INFORMATION AND THE EVOLUTION OF CODON BIAS

BY

DANIEL T. WRIGHT

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Library and Information Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Doctoral Committee:

Professor Les Gasser, Chair
Professor Allen Renear
Professor Zan Luthey-Schulten
Assistant Professor Miles Efron

Abstract

The informational properties of biological systems are the subject of much debate and research. I present a general argument in favor of the existence and central importance of information in organisms, followed by a case study of the genetic code (specifically, codon bias) and the translation system from the perspective of information. The codon biases of 831 Bacteria and Archeae are analyzed and modeled as points in a 64-dimensional statistical space. The major results are that (1) codon bias evolution does not follow canonical patterns, and (2) the use of coding space in organsims is a subset of the total possible coding space. These findings imply that codon bias is a unique adaptive mechanism that owes its existence to organisms' use of information in representing genes, and that there is a particularly *biological* character to the resulting biased coding and information use.

For my family, without whose love and support none of this would have been possible. Or even begun!

Acknowledgments

This project owes its existence to many people. First, I would like to express my deep thanks and appreciation to the faculty and staff of the Graduate School of Library and Information Science for many years of support and encouragement. I have never worked in a place as welcoming, collegial, and genuinely interested in oddball ideas, and I will be very lucky if I am ever able to do so again.

My great thanks to Allen Renear and Miles Efron for being so gracious as to serve on my committee, and for their most helpful insight and guidance in developing the present work.

Thanks, too, to my adviser, Les Gasser. I owe Les a great debt for his advice, guidance, and belief in this project – without which I do not think I could have made it all the way through. Most importantly, his enthusiasm and excitement about this information-in-biology stuff was invaluable in helping me see where I was going and what it was for, even when the way forward became murky to me.

I have known Zan Schulten for many years, beginning before I made the decision to work on a PhD and have a go at this whole "science" thing. Thank you, Zan, for the opportunity to work with and learn from you, and for being an inspirational scientist – you set a great example for working on fundamental problems that are really important. I do not think I could have embarked on this path without your example and assistance.

I owe perhaps the greatest debt of thanks to Carl Woese. Without his great insight into the workings of evolution and the related questions that really *matter*, I would never have become interested in pursuing this mix of information science and biological theory. I owe

much to our long talks about the fundamental relationship between evolution, information, and reality itself. Such mentoring is rare and exceedingly valuable, and I feel fortunate to have benefited from it.

Finally, my great thanks to my family and friends – for encouragement and love and support, and for listening to me talk about this crazy business for years. I could not have done this without everyone in my life being just who they are, and my most sincere thanks to you all.

Table of Contents

List of Tables

List of Figures

Chapter 1

Information science and biology

The "modern" form of cellular life has existed on Earth for approximately three and a half billion years – a remarkably long time for the persistence of an essentially stable form. The most crucial juncture in the evolution of cells from a more primitive ancestral form was the development of a general information storage and representation mechanism. According to Carl Woese, an evolutionary biologist who has contributed much to our understanding of early life, "the most important of these [evolutionary] junctures...was the development of translation, whereby nucleic acid sequences became symbolically representable in an amino acid 'language,' and an ancient 'RNA-world' gave way to one dominated by protein" (Woese, 2002) – in other words, the evolution of a system of symbolic information was the single most important evolutionary juncture in the history of life on Earth. Given its foundational place in early evolution, it seems likely that understanding the complexities of biological systems will require understanding the ways they process and use information. Furthermore, these cellular information system are both collective *and* individual – they exist and have their primary effects at the level of individual cells, but they evolve and increase in complexity only through large-scale collective processes.

Information science is concerned with the description and understanding of systems of

information, and with their impacts on the world. The largest and most abundant systems of information are of natural origin: they are biological. There is an enormous degree of complexity in individual cells, but large-scale microbial communities are complex in ways that dwarf human-built information systems. This may seem like a surprising statement, given the vast scale and complexity of human-constructed and human-scale information systems. For example, the Internet has on the order of 10^9 nodes and the human brain has on the order of 10^{11} neurons, and in both of these systems these individual components are connected together in immensely complex, dynamic networks. However, both pale in comparison to the size and complexity of microbial communities. For example, oceanic microbial communities are of immense size – 10^{27} cells, and many more viruses – with complex cross-species activity in their gene networks (DeLong et al., 2006; Beja et al., 2000). Furthermore, the turnover time[1] in these ocean microbial communities is roughly 1.5 days;[2] this means that there are roughly 6.6×10^{26} genome replication events every day. Even using a conservative estimate of average microbial genome size of one million base-pairs, this implies that information transactions involving 1.32×10^{33} bits of information are occurring every day;[3] by comparison, it has been estimated that there were slightly fewer than 10^{22} bits of information processed on the entire Internet in 2010 (Short et al., 2010). Since mutations primarily occur during replication, this immense number of replication events means that

[1]The average time it takes for every individual in the population to be replaced.

[2]Lecture by Ed DeLong at the University of Illinois' Chancellor's Colloquium on Evolution, Feb. 17, 2010.

[3]Since each base has four possible values, it can be represented by two bits of binary information. Hence, $(6.6 \times 10^{26})(10^6)(2) = 1.32 \times 10^{33}$. Also note that this is a *minimum* estimate of amount of information available; in reality there is other information (regulatory, structural, and other kinds of hereditary information) that is harder to quantitatively estimate but that nonetheless means this estimate is almost certainly low.

"one-in-a-million" events leading to the development of new, useful genomic information are actually quite common, making these microbial communities generators of innovation and information on an overwhelming scale.

From the perspective of information systems, defining what we mean by "biological information systems" in a precise way remains problematic. Nonetheless, it seems that a thorough understanding of information or information systems must include some understanding of the biological examples all around us. The present work aims to clarify some of the questions surrounding biological information by exploring the coding properties of organisms' use of the genetic code. These results indicate that the ways in which organisms encode information is a subset of all *possible* ways they could encode that information, which suggests that there may be a specific character to "biological information systems" to be further explored and defined. However, I do not intend to advance a comprehensive theory of (or address all the controversies around) biological information at this time. I am conducting research that will contribute to an improved understanding of how biological information can provide insight and integrate explanations of biological phenomena, and that may lead to such a theory in future work.

1.1 Heredity, descent, and information

One of the most fundamental properties of biological entities is their ability to reproduce themselves with a high degree of fidelity. It is this capability that we recognize when discussing genealogy, phylogeny, relatedness, speciation, and so on. This ability is both funda-

mental to defining biology (Barbieri, 2002), and one of the primary areas of inquiry into how organisms work. Biological reproduction is not a matter of "replication"; oak trees only give rise to oak trees, yet it is clear that no two oak trees are identical. Rather, it is a *process* that gives rise to descendants that are like, yet not identical to, their parents. How this sameness-with-variation might function has long been of primary interest to many biologists – and though we now understand much of this relationship, it is by no means fully understood. At the root of it lies information: an observer's judgment of similarity and difference is an informational one; and the only identifiable thread running through a genealogical line, linking ancestors to their recognizably similar descendants, is an informational one.[4]

1.1.1 Material v. non-material heredity

One important distinction to be made in understanding issues of heredity and reproduction is the difference between physical material from an ancestor (the molecular contents of the germ cell), and inherited information (what those molecular contents represent). There is naturally some continuity of material substance: physical DNA molecules, cell wall structure, and ribosomes are a few of the important – crucial, even – material things passed from one generation of cells to the next. However, these particular physical things are not (necessarily) passed to the *next* generation – copies of them are. So, then, it is important to note that while continuity from one generation to the next can be physically traced, this does not extend to following generations. Despite this, descendants are just as capable of producing

[4]There is also a fundamental question about why is it that organisms reproduce themselves at all, and why do they do so with fidelity?

more cellular components and machinery as their ancestors were – and these components are "the same" as they were in their ancestors. This is an example of the kind of "sameness with difference" that information is arguably required for, and that is linked to informational processes.

1.2 The importance of an informational view

Why does the informational nature of biology matter? In other words, why can we not make do by explaining all relevant phenomena in terms of chemistry or physics? In fact, it has been argued by a number of authors that this is what we *should* do (Sarkar, 2000; Sterelny, 2000; Godfrey-Smith, 2000, for example). These approaches to understanding biology reject emergent/higher-order phenomena (like information) because they are considered unnecessary, and possibly confusing. The argument of these authors is essentially that explanations like information may be intuitively satisfying – and may even be truly useful as analogy – but that at base, "information" is an anthropomorphic concept that does not have any true explanatory power.

The primary consequence of rejecting information phenomena in biology is to require connecting many contingent and specific events to explain phenomena, which results in explanations and theories that are more difficult to generalize. This difficulty in generalization presents a problem because generalizability is one of the key desirable features of any theory – this is now a good theory allows us to make predictions and discover new facts. The approach I take, on the other hand, regards information as a primary part of biology and

hopes to provide a more general framework with which to understand a wide range of phenomena. It is suggestive (though by no means conclusive) that features like synonymy and representation appear at many levels in both biology and in human language and information systems.

Thus far, there have been few efforts to explain the behavior and evolution of a biological system from a primarily information perspective; I hope that doing so with one of the mechanisms of adaptation in the genetic code (codon bias, to be explained below) will help to bring more serious attention to the nature of biological information.

1.2.1 The gene

One of the least controversial places information exists in biology is in the genetic system, as even some of the critics mentioned above have observed (Sarkar, 2000; Sterelny, 2000; Godfrey-Smith, 2000). The major role of the genetic system is in heredity – the passing of stable traits from one generation to the next.[5] It is worth examining this function in some more detail, for heredity helps to reveal some fundamental properties of information and information systems.

It is not surprising that one of the least controversial loci of information in organisms is also one of the primary hereditary mechanisms: the gene. Before moving into a detailed discussion of the gene, however, it should be made clear that hereditary mechanisms are not limited to genes. Furthermore "the gene" as it is widely understood is a problematic concept.

[5]While this is not the only function of the genome, nor is the genome the only biological system to exhibit heredity, I am using only this example here for the sake of clarity.

In the first place, we now know there are many examples of heredity through mechanisms other than genes; these include stably-inherited acquired morphological traits in ciliates and plants (Nanney, 1968; Landweber et al., 2000; Laland et al., 2008), cell-wall lipid structure (Kandler and König, 1998), regulatory gene network states (Huang, 2009), and genomic methylation patterns (Martienssen and Colot, 2001). In addition, the idealized model of genetics (introduced by Mendel – more on this below) that has a single trait controlled in a linear way by a single gene or a small group of genes is not completely accurate. While it does hold for certain traits under certain circumstances, most phenotypic traits are determined by a complex interaction of many genes, regulation networks, and environment. This is especially true given today's understanding that the physical substrate of the gene (described below) is to be equated with "the gene" as unit of heredity, which has led to a widespread belief that genes create traits in a directly determined way. Despite these shortcomings, this sort of DNA gene is the subject of the present work because studying it can provide crucial insights into the way organisms and cellular processes encode and use information. Thus, it is important to understand the strengths and limitations of that viewpoint before proceeding.

First conceived long before its physical realization was discovered, or even before there was a clear idea how "genes" might work, the notion that there must be some material that "programs" development and heredity was acknowledged by biologists. Darwin and some of his contemporaries speculated about "pangenes", which were understood to be hereditary particles of some sort that migrated from the body to the reproductive cells. It is clear even

from this early conception of the pangene that it must have had an informational component. Indeed, since hereditary traits (like eye color in humans or leaf shape in plants) were clearly passed down from parents to offspring, it would be hard to imagine otherwise.

The specific ways genes function were clarified by Gregor Mendel in the mid 1800s, whose famous experiments with peas established the "rules" of recessive and dominant genes and provided a clearer account of how genes act as information carriers (Mendel, 1866). The crucial insight of Mendel was that genes have discreetness; despite the fact that the rules of Mendelian genetics are a simplification of the complexities of real biological development and inheritance, they are accurate in their demonstration of the largely digital nature of heredity. The actual word "gene" (in addition to the related and important concepts "genotype" and "phenotype") was coined around 1910 by Wilhelm Johannsen.

In the 1940s, the material substance of the gene was found to be deoxyribonucleic acid (DNA) (Avery et al., 1944), the structure and digital-coding nature of which was later discovered by Watson and Crick (Watson and Crick, 1953b,a). The solved structure of DNA answered questions about its manner of replication, and shortly thereafter the triplet code establishing the binding between nucleic acid sequences (in genes) and amino acid sequences (in proteins) had been discovered (Ochoa, 1963; Nirenberg and Matthaei, 1961). With the discovery of the genetic code and the gene's method of replication, genetics was thought to be a solved problem; however, this was mistaken, as very little had been understood about the whole system in which the genetic code operates – without whose interpretation machinery the code would be meaningless. There are many complex macromolecules and processes

involved in the process of creating and maintaining the genetic code, and understanding these and the ways they interact to form the information system of the code is crucial to our understanding of evolution and biological processes.

1.2.2 The system of the genetic code

The particular biological information system I am studying is the genetic code; specifically, the dynamic, adaptive properties of the genetic coding system. The genetic code is the mapping between nucleic acid triplets (in genes) and individual amino acids (in proteins). A codon – a triplet of nucleic acids, representing one amino acid – is composed of three positions each of which can have four possible values; therefore, there are sixty-four (4^3) possible codons. These codons are used to represent the twenty biological amino acids[6] and a STOP control signal, giving a total of twenty-one encoded possibilities.

The genetic code is a vital example of a biological information system. Significantly, it is found absolutely *everywhere* in existing organisms, with very little variation. In addition, genomes have been extensively sequenced and cataloged, and as a result there is a massive quantity of genomic data available. The universality of the genetic system combined with the quantity of data about that system makes it both amenable to study and of wide-ranging impact and interest.

The genetic code is not a relatively simple static mapping, however. In reality, it is the product of a complex, dynamic system and its use changes depending on a gene's evolutionary

[6]There are many more amino acids, but only twenty are commonly used in organisms and represented by simple codons. Two uncommon amino acids – pyrrolysine and selenocysteine – are also found in some proteins, but are not coded for directly.

history and an organism's present circumstances. This does not happen through changes to the mapping from codons to amino acids, but rather through other mechanisms that exist due to the particular structure of the code. Codon bias – the subject of the present work – is one of these.[7]

1.2.3 Codon bias

Given the redundancy structure of the code – roughly three-fold more codons than encoded possibilities – there are several ways we could imagine the code table being arranged. First, the codons might be evenly distributed; that is, each amino acid would be represented by the same number of codons. Another possibility is that not all the codons would be used: each of the twenty-one encoded objects would be represented by a single codon, and the rest of the codons would be unused (they would not translate to anything). A third possibility is much like the second, except that all the "unused" space in the code would instead represent a single encoded object – that is, one amino acid would have maximum possible redundancy and the others would have none. The last possibility is that all codons would be used, but the redundancy would be unevenly distributed – some encoded objects would be represented by more codons, and some by fewer. This last possibility is how the code is actually structured: redundancy is not distributed evenly in the table, in the sense that some amino acids are represented by multiple (up to six) codons and some by only one. Nor is use of the available redundancies distributed identically in actual genomes. Again, there are several possibilities; first, it could be the case that for an amino acid encoded by four codons, each codon would

[7] A theoretical model of genomes characterizing them in terms of codon bias will be presented in Ch. 3.

comprise about 1/4 of the total encoding of that amino acid – synonym use would be evenly distributed. Alternately, a single synonym could be chosen in each organism, and that one would be used exclusively. However, what is actually seen is somewhere in between: each synonymous codon[8] has different use patterns, with one codon being found predominately and the others used much less heavily while still being present to some degree. Moreover, the preferred codon differs between organisms. Thus, we have a puzzling information question: how and why do these codon biases arise, and what is their function?

1.3 Research questions

I am addressing the following questions in the present work.

1. **How did/does codon bias evolve?** This question is broken down into three smaller questions below, which taken together will provide a picture of the evolution of codon bias.

    i. **What is the actual distribution of codon bias in sequenced genomes?** This data is needed to perform the analysis proposed below.

    ii. **What patterns of distribution may be seen on the canonical tree of life?** Analyze the data produced above by assigning bias values to nodes on the universal tree of life (see Ch. 2), and see if clusters of similar biases emerge. This will inform us about the relationship between evolution of the cell's central

[8]Two or more codons are synonyms if they code for the same amino acid.

information-processing machinery (used to build the tree of life [ToL]), and the evolution of codon bias.

iii. **Is the coding bias data rich enough to build a tree of bias evolution?** If so, the resulting tree may be compared directly with the universal ToL. If not, that result will tell us that the evolutionary dynamic of codon bias does not follow conventional genealogical patterns.

2. **What is the explanation for codon bias evolving this way?** In answering the questions above, we will have arrived at some understanding of the dynamics of codon bias evolution; answering this last question will serve to put those dynamics in an explanatory framework.

1.4 Structure of the thesis

The next chapter will address the existing body of literature on biological information and attempt to arrange it in an orderly framework, and I will present some theoretical arguments for the primary nature of information in biology. The rest of the thesis will then describe the structure and results of a case study to explicate one of the major systems of biological information: the problem of biased codon use in the genetic code.

1.5 An aside: general properties of information systems, and of information systems in biology

It is generally accepted that systems we recognize as being informational share certain properties. While there is no single definition of an information system, representation and coding, communication, and stability across time and space are all features of information and information systems. See particularly Dretske (1999) for a detailed account of how information necessarily must have these properties. Furthermore, it seems clear that the *sort* of information present in biological systems is the "information-as-thing" of Buckland (1991), as opposed to "information-as-knowledge" or "information-as-process." Despite the fact that processes are actually crucial to biological information, biological information is not Buckland's information-as-process because the physical processes of biology are not the same as the process of *becoming informed* to which Buckland refers. This is supported by Bates (2006), who begins with a foundational definition of information as "[a] pattern of organization of matter and energy" and proceeds to construct a naturalistic account of the fundamental kinds of information.

First, representation is crucial in systems considered informational. Representation is in many respects similar to coding: essentially, it is one thing standing in for some other thing. There is a close relationship between representation and coding, but representation is the more general property – all codes require representation, but not all representations are codes. One important and common use of representation is the mirroring of an external world in the internal state of an information system. This is seen in artificial symbol systems, computer systems that model the world, natural language – and very importantly, this kind

of representation is extensively seen in organisms' ways of interacting with the world through sensory data.

Second, coding is a foundation of many systems of information, especially communicative ones. Coding is best understood as a process of some symbol sequence standing for another, in some systematic way. Significantly, coding frequently provides space and/or time compression: that is, a coded message may take up less space then its decoded translation. Coding is seen in biology in the genetic code, where triplets of nucleic acids stand in for single amino acids. Human codes are abundant: diverse examples range from ASCII encoding of letters to linguistic coding of concepts.

Finally, persistence across time and space is an essential feature of information. Generally, the value of information lies in this feature: communication relies on being able to move a message from one place to another. Libraries and archives rely on their ability to encode information in a stable form so that it can persist for years or centuries, and networks of computers operate by shifting messages very quickly (but in finite and definite time intervals) through space. Organisms rely on time/space persistence most notably for mechanisms of heredity: the ability to pass stable information states to descendants. The most commonly known of these is the genome, but there are many others, including fixed/stable regulatory states and cell wall structures.

Chapter 2

The genetic code

The genetic system, and the code that it implements, is the most clear-cut example of biological information (Godfrey-Smith, 2000). However, our understanding of the evolution of the genetic system and our ability to explain how it came to be an informational system (in addition to being a mechanical and chemical one) are still incomplete.

2.1 Understanding the genetic system

First, it is important to have a clear understanding of what genes, the genetic code, and proteins are and how they are related. In the context of this work, a "gene" is a nucleic acid (specifically, deoxyribonucleic acid [DNA]) polymer chain with a particular sequence that encodes the amino acid sequence of a particular protein. This is, in fact, a particular kind of gene – there are also genes that encode other nucleic acid sequences (in ribonucleic acid [RNA]), and non-genic regions that play important roles in regulation and structure of the genome. This definition is in some sense subordinate to the more general (and much older) sense of gene: the stable association between some unit of hereditary information and an expressed phenotype (Carlson, 1991). However, the problem of codon bias applies specifically to protein-encoding nuclear genes, so this discussion will be focused there.

2.1.1 The code table

The "genetic code" is, broadly, the correspondence between triplets of nucleic acids in genes, and single amino acids in proteins. Genes encode a protein sequence, and the genetic code (as embodied in the translation machinery) determines how proteins are to be constructed from those gene sequences. The "universal" code table (figure 2.1) shows how these correspondences are set in virtually all organisms; the few exceptions to the universal table are minor and rare. While interesting in their own right, these exceptions (found primarily in mitochondria and some families of bacteria) do not need to be considered separately for the present study.

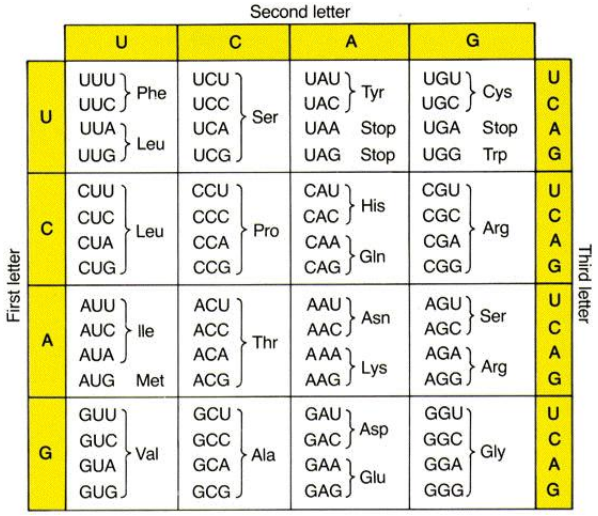

Figure 2.1: The genetic code table.

2.1.2 A brief narrative of the DNA → Protein process

A description of the process by which genes are translated into proteins will help to ground the following discussion of the mechanisms implementing this process. Figure 2.2 provides a graphical overview[1], and much of the following discussion is drawn from (Cooper, 2000, ch. 7).

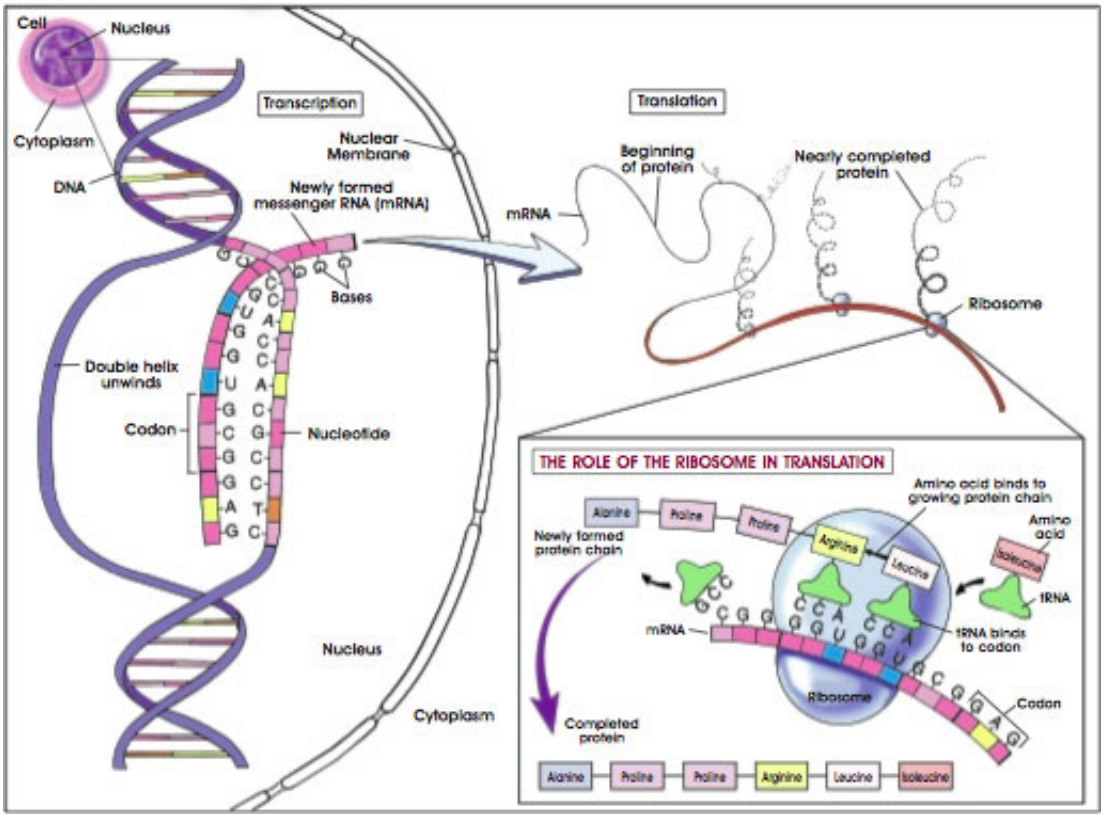

Figure 2.2: An overview of the transcription/translation process. From http://stemcells.nih.gov/info/scireport/appendixa.asp.

First, a DNA-dependent RNA polymerase "unzips" the DNA double-helix structure and assembles an RNA copy – called "messenger RNA" (mRNA) – of the DNA gene. This step is

[1]In addition, the reader is referred to http://www.youtube.com/watch?v=u9dhO0iCLww for an accessible and entertaining overview of the translation process.

called transcription; it is important to note that transcription means that the coding gene and the protein being assembled never interact directly – there is an informational intermediary that carries the sequence of the gene to the next stage, where the code is actually translated. This intermediary is informational in the sense that it embodies the same message as the gene it is copied from, despite being a different physical material. Even though there is no coding process at this stage, information is necessary in this movement of a coded message from one place to another in the cell – it facilitates displacement in space and time.

Next, the mRNA sequence is translated to a protein sequence by a ribosome. Where in the cell this takes place varies: in Bacteria or Archaea, translation may begin before transcription is even finished; in Eucarya, the mRNA may undergo processes of splicing and editing (in which its sequence will be rearranged to produce a protein product different than what the gene codes for directly) before it is translated. However, the fundamental process (translation at the ribosome) is the same in every organism. Many other molecular components are also involved in the translation process: tRNA, elongation factors, and aminoacyl-tRNA synthetases all play crucial roles and are aided by the ribosome. In the next section, we turn to the specific machinery that participates in the process outlined above.

2.1.3 Code machinery

Since the genetic code is universally distributed, it follows that the cellular mechanisms that implement the code should be universally distributed as well. This is, broadly, true; however,

each of the three domains of life (the largest divisions in the universal phylogenetic tree) has its own version of the core genetic information-processing machinery. Each of these versions is more alike than different; the incompatibility of parts derives from great specialization of these systems, rather then drastic differences in function. Therefore, the following general discussion of the machinery and its function is applicable to all three domains, except where otherwise noted.

There are four main cellular components responsible for turning DNA genes into amino acid proteins: RNA polymerase, the ribosome, tRNA, and aminoacyl-tRNA synthetases (aaRS). Of these, polymerase and the ribosome function as processing machinery, and tRNA and aaRS are responsible for actually setting the code – that is, they establish the correspondence between genetic triplets and amino acids in the final protein product.

Processing machinery: polymerase and the ribosome

DNA-dependent RNA polymerase is a large protein complex, responsible for unzipping the DNA gene and assembling a complementary copy in RNA. The product of RNA polymerase that we are concerned with here is mRNA; however, polymerase is also responsible for producing non-translated RNAs like ribosomal RNA, micro RNAs, and ribozymes from RNA genes. The process by which RNA polymerase works is made possible by the structural information inherent in DNA: a DNA gene is composed of two complementary strands (called "sense" and "antisense" depending on which is being read) in a double-helix structure, and when the DNA structure is opened by a polymerase the sense strand will guide the assembly of its compliment.

The ribosome is a large macromolecular structure composed of both RNA and proteins, and it serves to support the mRNA and charged tRNAs during the translation process. In this way, the ribosome isolates the translation center from thermodynamic noise and facilitates the proper mechanical arrangement and sequencing of the process; however, it does not play a role in setting or enforcing the genetic code itself.

Setting the code: aaRS and tRNA

The molecules responsible for actually setting the genetic code are the transfer RNA (tRNA) and aminoacyl-tRNA synthetases (aaRS). Each is responsible for one side of setting the code: the tRNAs recognize the anticodon (the three base complement of the gene sequence being translated, which is found on the mRNA) and placing the corresponding amino acid in the growing protein chain, and the aaRS charges the tRNA with the correct amino acid. In this sense, the aaRSs are ultimately responsible for the code: if they mischarge the tRNA the produced protein will not match the canonical code. This may be seen clearly in experiments that modify the aaRSs and the tRNAs to introduce novel (non-natural) bases into the translation process (Hausmann et al., 2007; Wang et al., 2009).

Other components of translation and transcription

A number of other large proteins play roles in translation. These include elongation factors[2], chaperonins[3], and mRNA editing complexes[4]. While none of these play a direct role in the matters the current work is concerned with, they are nonetheless an important part of the evolutionary story of the whole genetic system. In part, this is because they serve to demonstrate the complexity and specialization required of the modern translation apparatus for maintaining its level of accuracy and speed.

2.2 Codon bias

Codon bias is the preference of a given organism's genes to use one synonymous codon over others in representing a given amino acid. There are a number of mechanisms and specific forms these biases take; they are reviewed below. I propose using codon bias as a way to understand the dynamic information system character of the genetic code, because codon biases are rapidly evolving adaptations of the code to particular environments and reflect the dynamic balance between coding accuracy, speed of translation, and availability of amino acids and other necessary materials for assembling proteins.

Changing codon bias is one of the few areas that modern organisms are free to "experiment" with the code and coding machinery. This is because modern organisms are tightly

[2]EF-Tu stabilizes charged tRNAs for transport to the ribosome, and EF-G plays a role in amino acid chain termination.

[3]Which help the growing polypeptide chain to stabilize and fold properly.

[4]mRNA editing is performed after transcription but before translation; examples include alternate splicing of mRNA, which is used to make different proteins from a single gene, and polyA tail addition.

integrated wholes, and changing the large-scale code machinery would "break" (or at least, drastically reduce the efficiency of) protein translation, either killing the organism outright or rendering it non-competitive. Given these constraints, changing codon preferences is the way that organisms can optimize or change use of the code without large-scale or catastrophic repercussions.

Codon bias has been a topic of much interest to biologists, and a great deal of research has been done to explain the mechanisms that cause it. Despite this, there has been very little work done to understand the big picture – that is, we understand how codon bias happens in certain circumstances, but we do not have a clear picture of why. We would like to understand the specific causes and effects of biasing mechanisms in the context of the whole dynamic coding system that exhibits this quality.

2.2.1 Specific kinds of codon bias

There are several related but somewhat different phenomena that may be called "codon bias." First, we may be talking about average codon bias, that is, the distribution of bias across an entire genome. Average codon bias will play a large role in a related measure – the G+C/A+T content of a genome. The main way for either of these to change is though synonymous mutations of particular codons. As a result, changes in codon bias drive changes in G+C content and vice versa; these cannot be meaningfully described as separate phenomena, though there are cases where it may be productive to talk about one rather than the other. The average bias of each codon is the measurement that will be used most

in my work.

Next, we may mean the codon preferences of a particular gene. These can be indicative of translation speed, or conditions under which a given gene may be up-regulated. If a gene has codon preferences that differ from the genome-wide average, it may indicate (for example) that a given gene is expressed under conditions of nutrient starvation (Ermolaeva, 2001). What this indicates is that the codon preferences in a particular gene may reveal things about an organism's mode of life or preferred environment, or conditions under which two genes with redundant function may be alternately expressed.

We may also talk about the distribution of bias along the length of a particular gene, or average distributions along the lengths of several genes. This measurement often exhibits interesting variation, though the reasons for this are less clear than in the cases above.

Codon bias may also be broken into three categories, as in (Carbone et al., 2004). These categories are *content bias* (overall biasing of codon counts – what is being measured in the present study), *translational bias* (biasing of codon usage to influence rates of protein translation), and *strand bias* (different bias found on leading and lagging strands of a genome). These categories are useful ways to think of bias, as they describe many causes of – or influences on – the codon bias of a genome.

2.3 The evolution of the genetic system

Much attention has been paid to understanding the evolutionary history of the genetic code. This evolutionary history presents a conceptual puzzle: the code must be able to

accommodate innovation and adaptation to the environment in order for organisms to be adaptable, yet it must also be conservative – it must be stable enough that organisms can keep growing and reproducing successfully. There are two aspects to this conservatism: first is a constraint imposed by complexity – in a sufficiently complex system, changing any one part may cause the entire system to collapse, and the modern genetic apparatus is certainly a system complex enough for this to be a problem. Second is a selection constraint: even if changes to the code are not catastrophic for an organism in an immediate sense, the efficiency of the existing machinery is so great that any changes are likely to result in a slowdown in reproduction rate, so that over several generations these changes are very likely to disappear from the gene pool.

The most widely known theory explaining the code's universality is the "frozen accident" of Crick (1968), which states that the code has the particular nucleotide triplet to amino acid mapping that exists due to historical accident, and that the code is unchanging in modern organisms because any change would have catastrophic consequences in the context of the long and complicated proteins found in modern organisms. While this observation of Crick's is true in a sense, it does not *explain* anything about the evolution of the code – even if it is a "frozen accident," that tells us nothing of its evolutionary history up to the point that it became so. As such, it is not really a theory of the evolution of the genetic code at all; it is an observation about the complexity of modern organisms.

More recent work has been done to investigate the actual dynamics of code evolution. As stated by Freeland and Hurst (1998), the modern genetic code is "one in a million" with

respect to several measures of optimality and error resistance. These measures include a minimal likelihood that a single point mutation will result in a different amino acid being introduced into the gene being encoded, and optimality with respect to the polar requirement – a measure of chemical similarity between amino acids. The polar requirement measures the overall polarity of an amino acid, and amino acids with similar polar requirements have similar chemical properties (Woese et al., 1966). The code is optimal in respect to this measure in the sense that more-common mutations that result in encoding a different amino acid are likely to encode an amino acid of similar polar requirement, thereby minimizing the disruption to the chemical activity of the protein being encoded (Vetsigian et al., 2006). These conclusions call the "frozen accident" hypothesis into question – it is hard to believe that an accident would demonstrate that level of suitability to its purpose, and begs for investigation into why and how this would be so. Vetsigian et al. (2006) have suggested an explanation by modeling the relationship between horizontal gene transfer (HGT) in primitive populations and the evolutionary dynamics leading to the production of a "one in a million" code. This work provides actual insight into possible mechanisms of code evolution, and goes some way to explaining how the modern code (with its apparent status of "frozen accident") came to be as it is.

2.3.1 The origin of codons

Several theories have been proposed for the specific ancient evolutionary mechanisms that would have led to the associations between DNA (or RNA) triplets and particular amino

acids. The first type of theory proposes specific, stereochemical associates between RNA and amino acids that would have eventually, through processes of stepwise code expansion, given us the modern code. In this model, certain amino acids are physically correlated with certain nucleic acids; over time, this would lead from a direct physical coding to tRNA-AA relationships in a primitive (inaccurate) translation process, and from that to today's accurate and complex ribosomal translation process.

However, relatively little work has been done to explain the forces driving the genetic code as a system of information representation – why should there be a code at all, and how could it begin? To address this question, we will start with a more fundamental point: why should information evolve at all – in biology or otherwise? This area is not entirely unexplored: some researchers have indeed proposed explanations and evolutionary scenarios for early evolution of the genetic system.

2.3.2 Early evolution of the mechanisms of translation

Woese (2002) has proposed a model of early cellular evolution that attempts to explain how a modern (complex) code could arise from earlier, simpler, coding associations between nucleic acid and amino acid chains. In this model, translation and the ribosome evolved first, in the context of short RNA genes that were translated directly (without a transcription step) and imprecisely. At this time, the evolution of cellular structure would be driven largely by horizontal gene transfer (HGT), resulting in an environment of "innovation sharing" that would naturally drive the genetic system to a point of universality and optimal robustness

(Vetsigian et al., 2006). At this point, the evolutionary dynamic (and information dynamic) would shift to lines of vertical (genealogical in the modern sense) descent and the emergence of the three primary domains of life: Bacteria, Archaea, and Eucarya. This model provides an account of how informational complexity could bootstrap and grow, leading to the emergence of the modern, highly complex genetic information systems. However, only certain aspects of this model have been tested through computational modeling, and much of it remains speculative.

2.4 Where does biological information appear?

Biological information may be found almost anywhere we look in organisms. Cell wall structure, genes, protein structure, organelle distribution and structure – all of these things are information in the cell. The notion of Maynard-Smith (2000) that biological information is property of genes alone is incomplete; information also lies in the heritable structure of cells (Dose, 1994; Jablonka and Lamb, 2006). This can be seen in the continuation of structural properties from parent to child cell, despite the small amount of physical material that is actually transferred. It is also seen in some kinds of epigenetic regulation that can persist though gene copying, and be passed from parents to offspring. One example of this is seen in the heritable – but non-genetic – cortical patterns of certain ciliates (Nanney, 1968). In fact, many general regulatory mechanisms induce heritable state changes, from methylation patterns (copied along with the genome) to concentrations of regulatory signaling chemicals (passed along during cell division).

2.5 Phylogeny

One of the tools for understanding the evolution of biological information is phylogeny – the study of patterns of evolutionary descent. Understanding the pattern a given information system has evolved in is crucial to understanding the dynamic principles that led it to evolve in that way. Phylogenetic trees (fig. 2.3) are a representation of the evolutionary relationships within a group of organisms, displayed as a branching tree where the leaf nodes are organisms or groups of organisms, and the internal nodes are some point of common ancestry.[5]

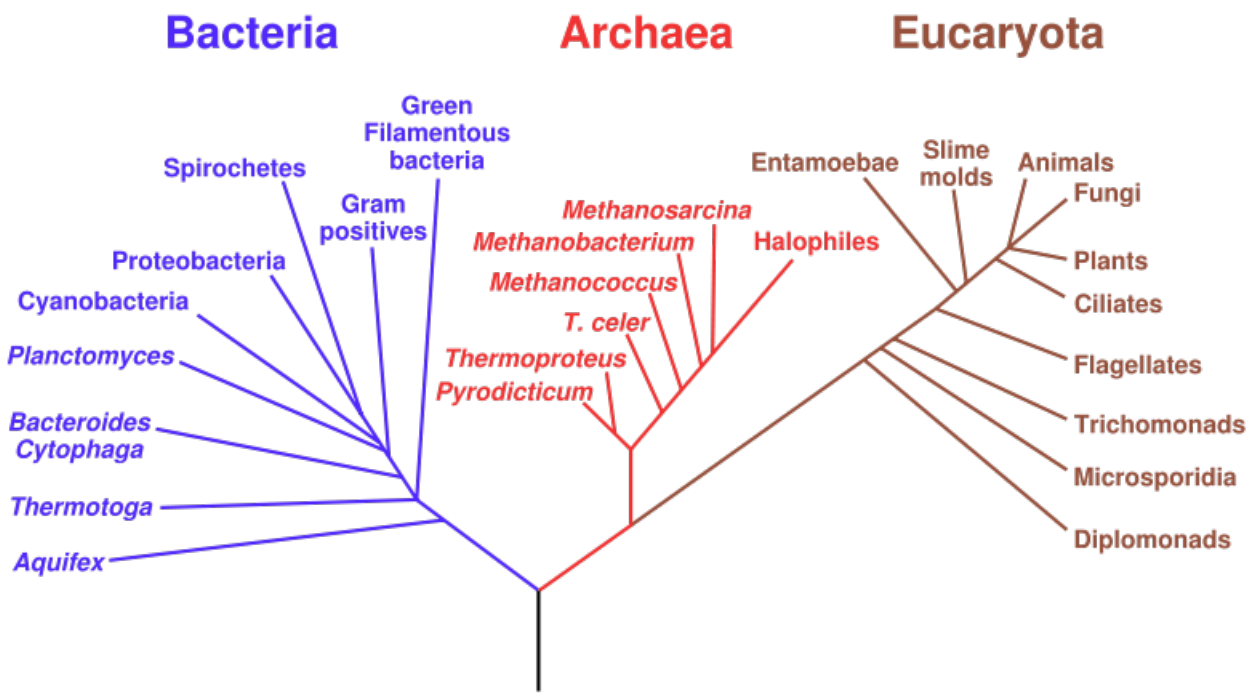

Figure 2.3: An example phylogenetic tree, showing the three primary domains and major divisions thereof.

[5]Possibly a specific common ancestor, but not necessarily.

The history of phylogenetics – the science of constructing and examining these trees – is long and colorful. For the purpose of the present work, it is important to understand that Woese and Fox (1978) began a line of work that has firmly established a universal phylogenetic tree widely thought to reflect the evolutionary history of all organisms using the sequences of their small ribosomal subunit sequences (commonly referred to as "SSU" or "16S" sequences). While there is some disagreement about the assertion that a tree so constructed represents a true evolutionary history, based on the argument that widespread horizontal gene transfer has completely obliterated *any* "true" phylogenetic record (Doolittle and Bapteste, 2007; Gogarten et al., 2002; Doolittle, 1999), those criticisms are of little concern in the present work. This is because I am concerned with the evolution of the genetic code, and the evolutionary history reflected by the universal tree is built using the central machinery that *implements* the genetic code; therefore, using this version of the phylogenetic record will provide correct results regardless of its true universality.

Chapter 3

A compositional-bias model of genomes

An aspect of the genetic code vital to any understanding of it as an information system is the uneven distribution of use among redundant codons, known as codon bias. The reasons for differing codon bias in different organisms are not well understood, though many possible reasons have been discussed. Qin et al. (2004) discuss these in detail; their primary reasons are rate regulation (controlling the speed of gene expression), environmental response (changing the relative expression levels of genes in response to nutrient availability), and horizontal gene transfer (newly-acquired genes and the host genome will equilibrate to a new, consistent codon bias). Each of these is seen in different species and under different circumstances. Indeed, many specific cases of codon bias are well characterized, but the deeper reasons for biases are not. Codon bias can be analyzed in a number of ways; these include looking at the global use distribution among codons, variation of bias along the length of a gene, and the very general measure of G/C (vs. A/T) percentage. Analysis of the codon bias phenomena has generally focused on biochemical considerations: physical stability, rates of mutation, conflicting evolutionary forces, and the like. However, this question may also be approached abstractly: in any coding system with redundancy (multiple symbols "meaning" the same thing), what leads to preference of one symbol over another? Naively, we would

expect a random distribution of synonyms, but we know that this is not what happens in the genetic system (see 1.2.3 for a more complete discussion).

I am studying the evolutionary information dynamics of the genetic code. This study will also contribute to understanding of dynamically evolving information systems generally by providing a case study of one such system with some general conclusions about how and why it exhibits the dynamic properties it does.

Using the model presented here, I have analyzed the average coding bias of 831 Bacterial and Archaeal genomes. This analysis leads to the conclusion that there is a "biological" region of codon bias space, which is a subset of the total codon bias space. Details follow in Ch. 4 and 5.

3.1 Codon bias space

First I will describe what is meant by "codon bias space," in the context of a general model describing genomes in terms of genome compositional bias. The data for the present study is a set of 1x64 vectors of codon *compositional proportions*. Conceptually, each vector describes a point in a 64 dimensional space – the "codon bias space." This space covers every *possible* configuration of codon bias; however, only a portion of the points in this space could possibly be occupied within the constraints imposed by the standard genetic code, and only a portion of *those* points are actually occupied by real genomes. By describing each genome as a point in this space, we can develop a spatial understanding of the distribution of points and the structure described by the data.

3.1.1 Details of the model

A codon is composed of three bases (x, y, z [eqn. 3.2]) chosen from the set B (eqn. 3.1). Each codon is assigned an index i, and each synonym group[1] is assigned an index s, in order to simplify discussion of codons and synonyms in the model (see table 3.1). The compositional proportion of a codon (cp_i, where i is the index of the codon being considered) is the number of times that codon is found in protein-coding genes divided by the total number of codons counted in its synonym group (eqn. 3.2-3.4). For example, TAT and TAC both code for Tyrosine (Tyr); in a genome with 3 TAT, 7 TAC, and 20 other codons, the matrix entry for TAT would be 0.3 and the entry for TAC would be 0.7. Normalizing codon frequencies to the size of their synonym group controls for systematic biases in the use of amino acids in different organisms; this sort of bias arises due to differing chemical properties of the amino acids, rather then through differences in coding preference. Each bias vector (Bv) contains sixty-four values, one per codon. Each species (or more precisely, each genome) is represented by one of these vectors (eqn. 3.5).

$$\{B : A, T, G, C\} \tag{3.1}$$

$$C_i = \text{count of codon } i \text{ in genome} \tag{3.2}$$

$$C_s = \text{count of all codons in } i\text{'s synonym group } s \tag{3.3}$$

$$cp_i = \frac{C_i}{C_s} \tag{3.4}$$

$$Bv = \{cp_0 \ldots cp_{63}\} \tag{3.5}$$

$$0 \leq cp_i \leq 1 \tag{3.6}$$

$$\forall s : \sum_{i \in s} cp_i = 1 \tag{3.7}$$

$$\sum_{i=0}^{63} Bv_i = 21 \tag{3.8}$$

[1]The set of codons encoding the same amino acid.

Index	Codon	Syn. group	Syn. index
0	ATG	Met	0
1	TGG	Trp	1
2	TTT	Phe	2
3	TTC		
4	TAT	Tyr	3
5	TAC		
6	CAT	His	4
7	CAC		
8	CAA	Gln	5
9	CAG		
10	AAT	Asn	6
11	AAC		
12	AAA	Lys	7
13	AAG		
14	GAT	Asp	8
15	GAC		
16	GAA	Glu	9
17	GAG		
18	TGT	Cys	10
19	TGC		
20	ATT	Ile	11
21	ATC		
22	ATA		
23	GTT	Val	12
24	GTC		
25	GTA		
26	GTG		
27	CCT	Pro	13
28	CCC		
29	CCA		
30	CCG		
31	ACT	Thr	14
32	ACC		
33	ACA		
34	ACG		
35	GCT	Ala	15
36	GCC		
37	GCA		
38	GCG		
39	GGT	Gly	16
40	GGC		
41	GGA		
42	GGG		
43	TTA	Leu	17
44	TTG		
45	CTT		
46	CTC		
47	CTA		
48	CTG		
49	TCT	Ser	18
50	TCC		
51	TCA		
52	TCG		
53	AGT		
54	AGC		
55	CGT	Arg	19
56	CGC		
57	CGA		
58	CGG		
59	AGA		
60	AGG		
61	TAA	STOP	20
62	TAG		
63	TGG		

Table 3.1: Codon indices and synonym groups.

Each cp_i must range between 0 and 1, since it represents the proportion of codon xyz in that genome (eqn. 3.6). Additionally, the sum of all values in Bv must be 21 (eqn. 3.8 – since each value in the vector is a proportion of the total composition of the genome normalized by the number of encoded possibilities), and each set of cp_i in a synonymous group must total 1 (eq. 3.7).

This model of codon bias reflects several deliberate decisions in interpreting the data. First, the bias of each codon is represented by the proportion of a codon within its synonym group, rather than its proportion in the entire genome. Second, I have chosen to include STOP codons and codons with no synonyms in the model, both of which are disregarded in many studies of codon bias. I believe that including these features reflect more fully the coding variations I am studying, and that these variations may help to distinguish evolutionary differences between genomes on the basis of coding variation.

3.1.2 Implications of the model

There are several conclusions we reach simply by analyzing the model and the nature of the genetic code. The basic form of the model describes any code where each symbol in the code can vary in frequency independent of the others, subject to the constraint that the total of these frequencies within each synonym group cannot be greater than 1 (eqn. 3.6). Given this, we conclude that each codon would have a cp_{xyz} of $\frac{1}{N}$ (where N is the number of codons in xyz's synonym group) in a hypothetical genome that has an equal proportion of each amino acid. In a sense, this "equal proportion" hypothetical genome

describes an "unbiased" genome, in the sense that it does not prefer any one codon over others within synonym groups. By comparing real genomes to this point, we can produce an abstract picture of how biased real genomes are, and their distribution in space around this "unbiased" point.

As we will see, only a subset of codon bias space is used by actual organisms, and this is significant – it describes a particular "biological" character within the total space of possible genetic coding.

Chapter 4

Composition-bias study of genomes: methods

4.1 A database of codon bias

4.1.1 Genetic data

The first step in analyzing the patterns of codon bias was to compile a database of biases for all the organisms in the analysis. The first step of this process was to download all data on protein-coding genes for Bacteria and Archaea from the NCBI Genbank genomes FTP site (found at `ftp://ftp.ncbi.nih.gov/genbank/genomes/Bacteria/`). Originally, data from the CUTG database of per-genome and per-gene codon counts (Nakamura et al., 2000) was to be used for calculating codon biases. However, once I downloaded and began to analyze the CUTG data, I discovered that it was lacked many organisms that were available in the NCBI data. More importantly, some of the organisms that were present were drastically incomplete, which threatened to skew the analysis. As a result, I decided to start from the raw NCBI protein-coding gene sequences, rather then using CUTG's preprocessed codon counts.

For each genome, all of the `.fft` and `.rpt` files available on NCBI's FTP site were downloaded; the `.ffn` file (or files – many organisms have several) contains the complete

sequences for each protein-coding gene in the genome (and plasmids), and the `.rpt` file contains useful information (like total sequence length and genetic code used) for each `.fft` file. From here, several data filtering steps were performed:

- any `.ffn` for plasmids (rather then core genetic data) were eliminated;
- redundant strains (strains from the same species) were eliminated;
- and organisms using a non-standard genetic code variant were eliminated.

Each of these steps removed data that would make the final analysis less clear (due to a confusion of influencing factors), or too difficult to perform in the context of the present project. However, each set of removed data sets (plasmids, multiple strains of the same organism, and non-standard codes) would make for an interesting codon bias study in its own right, and would be good subjects for follow-up studies.

4.1.2 Data processing

Following the filtering of the gene data, the raw gene sequences needed to be processed into counts of codons, and from there into the Bv vectors discussed in the bias model presented in Ch. 3. This was done using several Perl scripts I wrote for this purpose; please see Appendix C for a complete listing of these programs.

First, a database of the raw codon counts for each organisms was assembled using `ncbi_process.pl` and `make_counts_from_ncbi.pl`. The intermediate file format is the same as that of the codon count data from CUTG; since this study began using CUTG data, the next step in the process expected CUTG-format count files.

Next, the NCBI counts were processed into a database of codon frequencies using one of two scripts: `make_bias_counts.pl`, which compiles global compositional proportions for each codon, or `make_bias_counts_grouped.pl`, which compiles compositional proportions normalized within each synonym group. The latter (normalized) format of the data is what was actually used for further analysis, and for the results presented.

Characterization of the bias space and generation of distance measures was done using `R`. Distance measures were generated using R's built-in `dist` function. The hypothetical "even distribution" matrix and distances from each organism to this point were generated using R code in the file `codon-funcs.r`, also found in Appendix C. Additional R code was written to use the built-in `svd` function to compute the first k singular vectors for each Bv, and store this data in a new list of vectors suitable for analysis.

4.1.3 Limitations

The statistical methods used in this study are intended to be exploratory, not exhaustive. As such, testing conclusions against a null hypothesis (like that the patterns seen might be due to chance and not a systematic difference in distributions) has not been done, though less rigorous precautions (like looking at multiple distance measures and using multiple approaches to evaluate the shape of bias space) have been taken. While I feel confident in the results, a more rigorous and quantitative approach to bias space characterization would be a promising direction for future work.

In addition, there are alternate approaches using Shannon's mutual information and the

entropy of relevant distributions (Shannon, 1948) to describe the space. These would be used to measure the entropy of the space relative to a randomly-generated space, and could be used measure mutual information both within and between Bv. The picture of bias space provided using these information-theoretic approaches would complement the kind of statistical picture provided by the present analysis; however, they are outside the scope of the current study. A future study using information-theoretic measures to characterize bias space would be a valuable contribution and help to round out our understanding of this area.

Chapter 5

The biological character of codon bias distribution

5.1 The biological character of code use

Distances between points in codon bias space range from 0.00447 to 4.9289, with an average distance of 1.8085.[1] They are distributed as seen in figs. 5.1 and 5.2.

Also telling about the uneven distribution of distances is the distribution of distances around an arbitrarily-chosen "even distribution" point. This is the point in codon bias space occupied by a hypothetical genome with evenly-distributed[2] coding bias, chosen solely to provide a reference point. The distribution of distances relative to that point is quite uneven, suggesting (again) a non-uniform density of points in bias space are occupied by real genomes. A similar distribution is seen with both Euclidian and Manhattan distance measures, suggesting that this distribution is not an artifact of the distance measure chosen (figs. 5.3, 5.4).

The "lumpiness" of this space can also been seen from a plot of the first two singular vectors of each Bv (fig. 5.5). Due to the correlation of values within synonym groups, these

[1] In an abstract Euclidean space. These distances may be thought of as some kind of evolutionary distance, and are used to construct a UPGMA tree in Ch. 8.

[2] Meaning that, for example, an amino acid with four synonyms code $\frac{1}{4}$ of the occurrences of that amino acid with each codon.

singular vectors capture a great deal of the total variation in each Bv (roughly 65%).

It is also clear from this picture that there are two regions of bias space that are more densely populated – on the top and bottom left of fig. 5.5. This is in line with the findings of (Carbone et al., 2004), who found a clear division between thermophilic (and hyperthermophilic) and mesophilic organisms in bias space. While the present study does not address organism lifestyle as Carbone et al. did, a similar finding of a lumpy bias space is strongly suggestive.

5.2 Conclusion

Codon bias space is a useful conceptual model for understanding organisms' use of codon bias, and it furthermore facilitates analysis and visualization. As a result of developing and using this model, I have demonstrated that actual codon biases do not cover all of the possible area in bias space that they could. In addition, the density of distribution of organisms in the space that is covered is not even, suggesting that some regions of this space are preferable to others. This may be due to environmental and lifestyle factors as suggested by (Carbone et al., 2004), or it may be due to other factors such as biasing in a particular region being more amenable to rapid translation of particular proteins. In any case, the conclusion that there is a *particularly biological character to codon use* – that is, that organisms prefer a subspace of the total hypothetical codon bias space – is very significant. Furthermore, it suggests that biological information use in general may have particular characteristics that ought to be investigated.

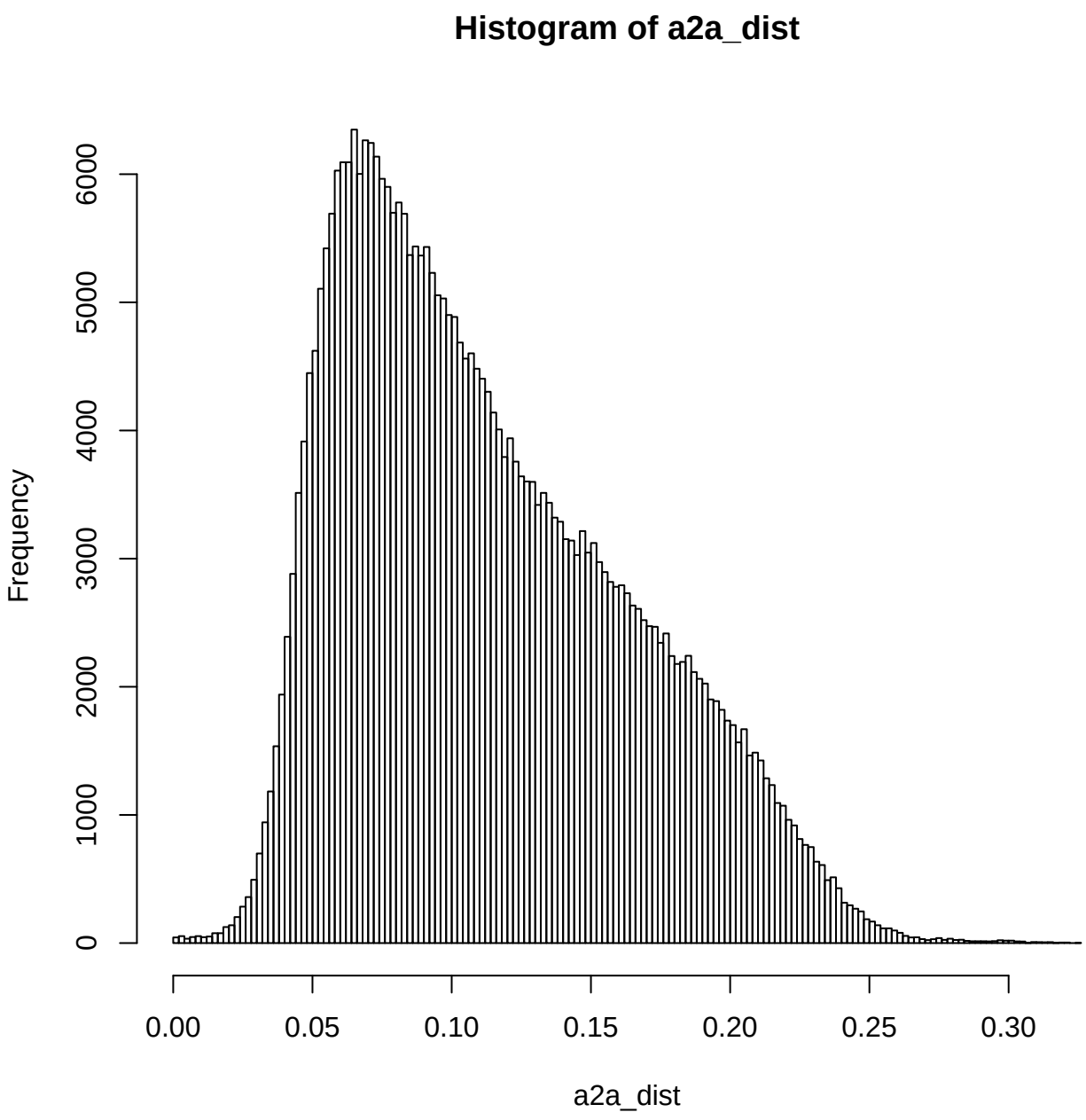

Figure 5.1: Histogram of all-to-all coding distances between 831 genomes.

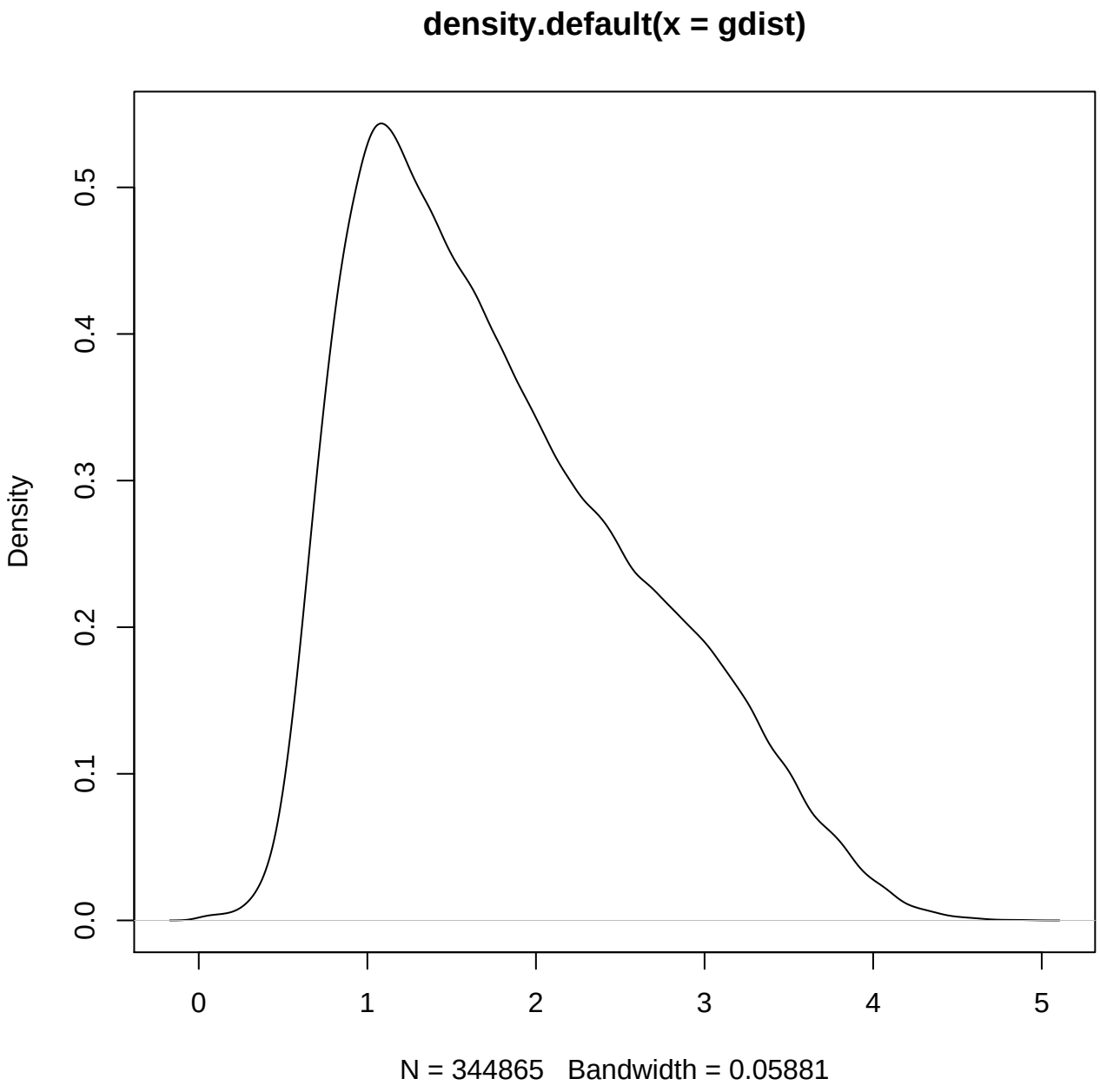

Figure 5.2: Density of all-to-all coding distances between 831 genomes.

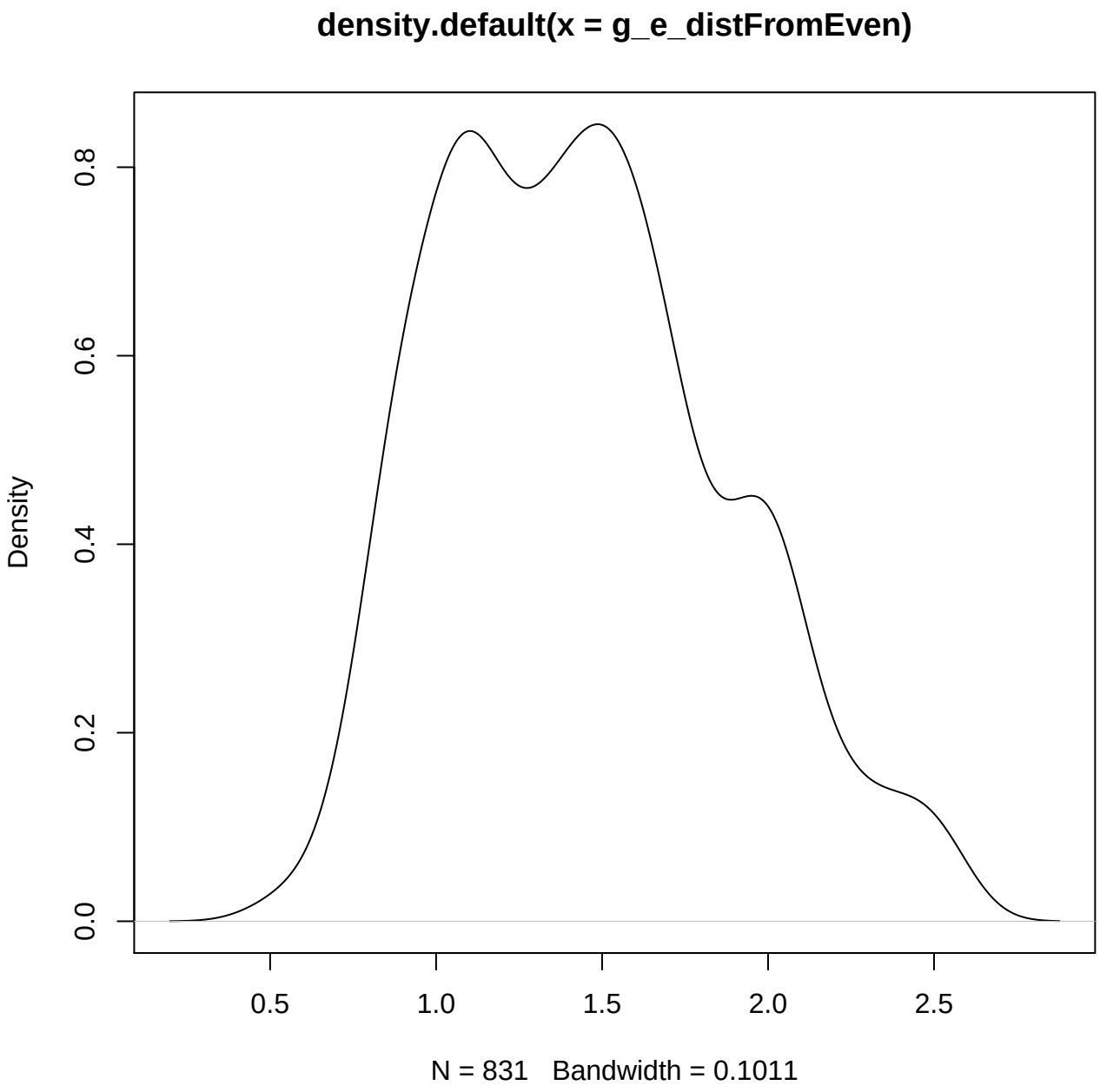

Figure 5.3: Density of coding distances from 831 genomes to point of even coding distribution. Euclidian (L2) distance.

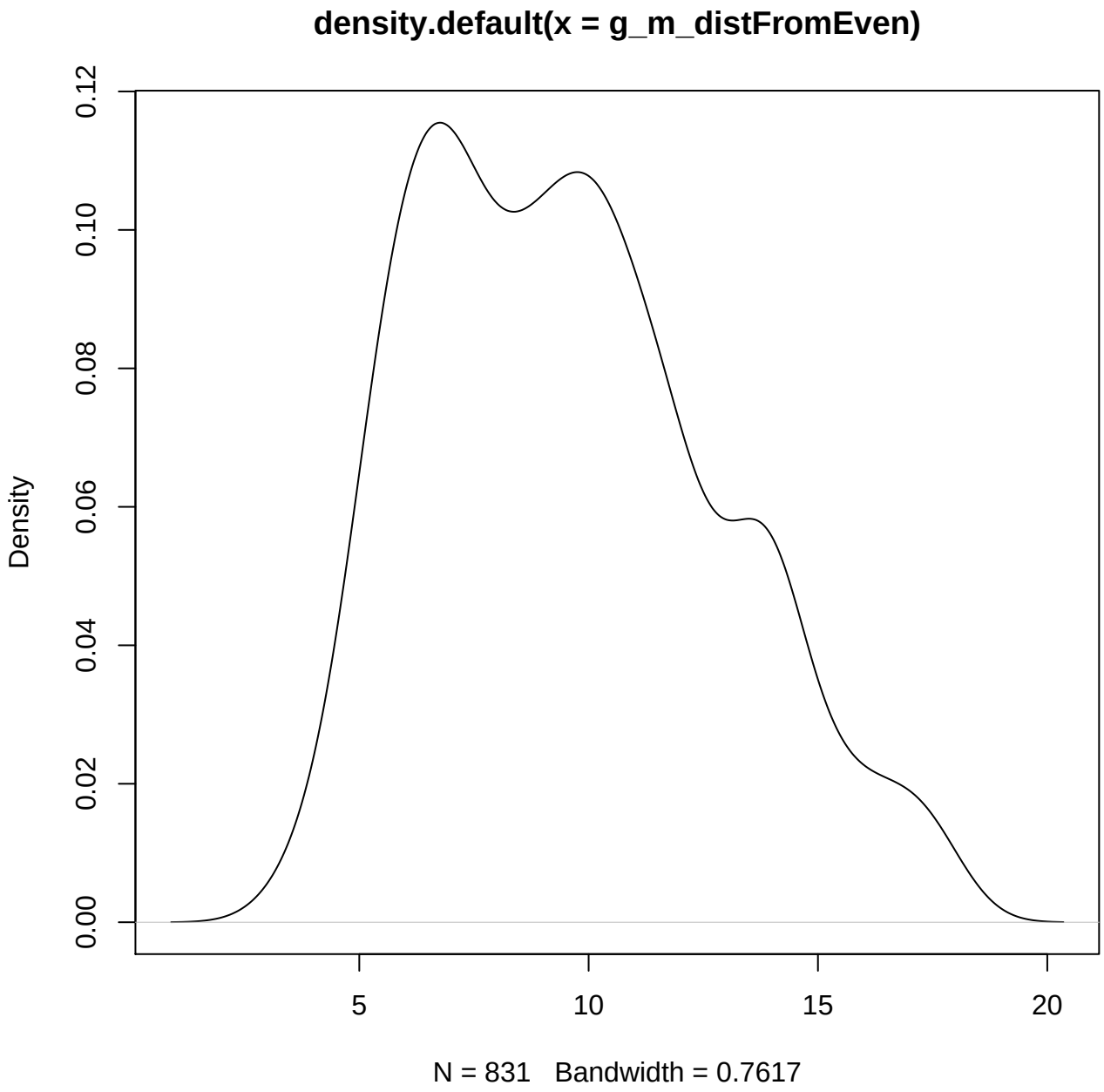

Figure 5.4:)
Density of coding distances from 831 genomes to point of even coding distribution. Manhattan (L1) distance.

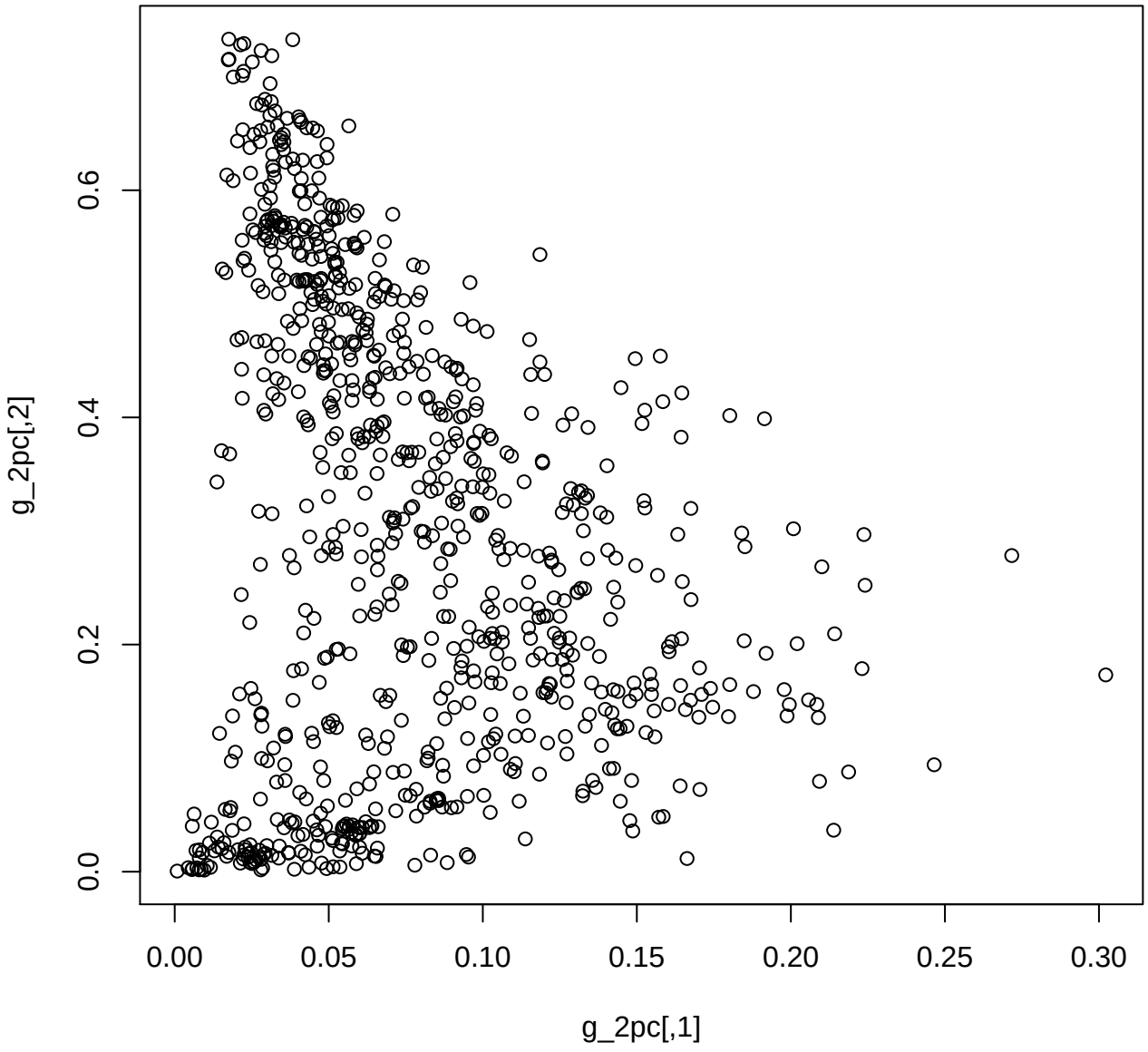

Figure 5.5: Plot of first two singular vectors for all Bv.

Chapter 6

The phylogenetic model of evolution

Using a phylogenetic framework to interpret biological data means assuming that the data describe a set of organisms descended from a single common ancestor via a sequence of intermediate forms, and that the differences in the data can be interpreted to reconstruct this hypothetical historical relationship. Methods for doing this using sequence data and phenotypic characteristics are well understood. I am focusing in particular on the universal evolutionary tree – the tree constructed from ribosomal sequence data, and thought to represent the true organismal evolutionary tree (Wheelis et al., 1992).

6.1 The universal tree

The universal tree is one constructed using 16S ribosomal subunit sequences. The 16S sequences are thought to be identical with the evolutionary core of organisms; therefore, interpreting them in a phylogenetic context is thought to produce an accurate historical account of evolution. However, the determination of the evolutionary core is problematic.

The difficulty in determining this "core" arises from the nature of genomes, especially in microorganisms: many genes are subject to widespread horizontal gene transfer (HGT) – as opposed to vertical descent along phylogenetic lines of inheritance. One outcome of

widespread HGT is that the genomes of many organisms are a patchwork of genes from other species and lineages, so that a "gene tree" – a phylogenetic tree constructed the sequence of a specific gene – may not be representative of the actual evolutionary relationships between the organisms on the tree. For example, the tRNA synthetases (aaRSs, mentioned above) – of which every organism has twenty, one for each amino acid – have been subject to widespread HGT which has completely obliterated the true evolutionary trace for some of them (Woese et al., 2000), while obscuring it or leaving it untouched in others. However, some genes – such as those for the ribosomal machinery – are thought to be immune from HGT in modern organisms, because of their central and extremely integrated position in the cell (Wheelis et al., 1992). The argument is that any change to these genes would be catastrophic, since protein synthesis is an extremely complex process and changes to the ribosome would disrupt it. This means that any organism with HGT'd ribosomal components would (at worst) be unable to manufacture proteins correctly (leading to death), and at best would produce proteins less efficiently – rendering that organism's descendants non-competitive. As a result, the ribosome is thought to represent the true genealogical trace when used to assemble phylogenetic trees. The 16S subunit of the ribosome was first used to do this because it was possible to handle using the technology of the time (Woese and Fox, 1978), and it has proved to be a robust target for phylogenetic analysis even as longer rRNA sequences and the sequences of many more organisms have become available (Roberts et al., 2008).

Due to the strength of the universal tree, it is being used as the baseline for evolutionary comparison in the present study.

6.2 Trees of bias data?

As stated above, the phylogenetic model is based on an understanding of evolution that assumes successive descent-with-modification from a common ancestor. As such, phylogenetic trees do a poor job describing and accounting for non-genealogical evolutionary forces, like horizontal gene transfer (HGT). In addition, the phylogenetic model depends on having enough data about terminal descendants of the evolutionary line – that is, modern organisms – to resolve each branching point (common ancestral state) between the common ancestor and the present. As a result, detailed data capable of distinguishing between many states is necessary when working with large numbers of organisms, if one is to arrive at a reasonable phylogenetic tree. In addition, the data must be generated by an underlying genealogical *process* in order for a phylogenetic interpretation to be valid, or even possible.

In the case of the codon bias data, neither of these conditions is established for certain. It may be the case – in fact, it is likely – that the 64-position bias vectors (the Bv of the bias model, presented in 3) do not contain sufficient information to distinguish amongst the 831 organisms being considered in the present study. Furthermore, it is well known that codon bias is influenced by factors like HGT; given this, it is unknown if the genealogical assumptions necessary to produce reliable trees will hold. As as result, constructing a bias data tree is being approached as an open question: if traditional phylogenetic methods fail, then using a simple distance based clustering approach will be tried. The failure of traditional phylogenetic methods will produce an interesting result in itself, as it tells us something about the lack of similarity between phylogenetic assumptions and the character

of codon bias.

Chapter 7

A phylogenetic study of codon bias evolution: data sources and methods

7.1 Tree construction

7.1.1 Construction of the universal tree

The tree built in this study uses 267 non-redundant 16s sequences downloaded from the Ribosomal Database Project (RDP)[1], processed using `dnaml` from the Phylip package (Felsenstein, 2005). A full set of all cultured 16s sequences for Bacteria and Archaea was downloaded from RDP, and all redundant strains were eliminated. Next, sequences for which a matching NCBI TaxID in the bias data could not be found were eliminated. Working with the RDP data required writing a number of Perl scripts to filter, process, and compare sequence metadata; these are [will be] listed in Appendix C.

7.1.2 Construction of the bias tree

The first attempt at building this used 64 position feature matrices (one matrix per organism) input to PHYLIP (Felsenstein, 2005), a widely-used phylogeny calculation program. These matrices are identical with the Bv for each organism, and each position in the matrix was

[1] http://rdp.cme.msu.edu/

treated as a continuous phenotypic character for PHYLIP's `contml` program. This program uses a maximum-likelihood approach to calculating phylogenies. Alternative techniques are maximum-parsimony and Bayesian methods; maximum-likelihood is preferred in this case because it makes fewer assumptions about the data than maximum-parsimony, and Bayesian methods are considered controversial (Felsenstein, 2004). Significantly, tree building using `contml` failed; see Ch. 6 above for theoretical background on why this would be the case.

Next, an NxN distance matrix between all organisms was constructed in R, and used as input to R's `hclust` function. Using `hclust` in the average-linkage mode is identical with the standard UPGMA phylogenetic methodology, so the resulting dendrogram is interpretable as a phylogenetic tree. However, it is very important to keep in mind that the underlying processes leading to codon biases may not conform to phylogenetic assumptions! This means that while the tree does depict *something* valid and interesting about the relationship amongst different organisms in terms of codon bias, that relationship may not be genealogical.

7.2 Tree comparisons

Several approaches have been used to compare these trees. First, the trees were processed using the Perl script `compare_trees.pl` (Appendix C, which loads both trees into memory and compares the nearest neighbors of each terminal node in both trees. This gives a rough sense of end-state similarity – that is, disregarding the ancestral topology in each tree, do organisms end up in a similar final relationship to each other? Next, a scaling analysis

of the trees was performed, using the tool provided by TORNADO (Sipos et al., 2010).[2] This analysis provides a quantitative and visual analysis of tree topology by comparing node branch size (the length of the branch leading to a node) and the cumulative branch size for the same node (the length of all branches in the node's subtree).

Originally, I had intended to also do a more detailed topological analysis of the trees using some topology-analysis packages – Nye's Metatree method (Nye, 2008), which provides a "tree of trees" for comparing differing topologies, and *cousins*[3], a software package that produces a numerical "cousin distance" for scoring the similarity of trees. Interestingly, both of these software packages proved unsuitable for processing trees larger then a couple dozen organisms, and so had to be abandoned. A detailed topology analysis remains as an interesting topic for possible future work.

7.2.1 A note about tree comparisons

It must be noted that the methods and approaches used for comparing trees in this work are necessarily rough and imprecise. This is due in part to the state of available software in the field, but more importantly due to the complexity of the problem – comparing tree topologies and reaching precise conclusions is far from a solved problem. Furthermore, as these trees are fairly large, comparing them with existing techniques is computationally challenging. As a result, my analysis methods are intended simply to give a rough (but accurate) picture of whether or not these trees are alike. Precise conclusions in this area must wait for future

[2] http://tornado.igb.illinois.edu/scaling.html
[3] http://cs.nyu.edu/cs/faculty/shasha/papers/cousins.html

development of software and methodology.

Chapter 8

Phylogenetic results

Using the methodologies described in Chapter 7, I have compared the results of phylogeny construction using the 16S (small subunit) ribosomal RNA and the codon bias data. The results confirm the hypothesis that the pattern of codon bias evolution does not follow the same pattern as 16S evolution – except at the leaves of the tree. The alternate hypothesis – that bias evolution *does* follow a similar pattern to 16S evolution – is disproved conclusively.

8.1 16S tree features

The 16S tree constructed in the present study (figs. 8.1 and A.1) is essentially the same as the canonical 16s-based tree constructed by other authors (Pruesse et al., 2007).

8.2 Bias data tree features

The tree of bias data (figs. 8.2 and A.2) is constructed using a UPGMA (Unweighted Pair Group Method with Arithmetic Mean) approach on a matrix of Euclidean distances between each organism in codon bias space. Since this method works purely on a set of distances, it is able to construct a tree. More complex phylogenetic methods – like maximum likelihood,

treating each position in Bv as a continuous character in a feature matrix – fail to construct plausible trees. This is a finding in itself, and confirms a suspicion I had when beginning this project.

As a result, trees constructed using an ML approach do not have enough branch length to reliably discriminate between evolutionary scenarios. On the other hand, since UPGMA is a very simple clustering approach depending only on a distance measure, it is able to construct a tree; the problem of whether or not that tree is meaningful becomes a problem of whether or not the distance measure is trustworthy. Given the model of 64-dimensional bias space described above – from which the distances are drawn – this UPGMA tree does provide an accurate reflection of the relatedness of these genomes in that space. Inferring a great deal about literal evolutionary scenarios – in the sense that two organisms sharing some part of the tree have literal common ancestors in bias space – is not warranted, since distance in bias space does not have enough detail to accurately describe common ancestors. However, it is reasonable to assume that internal nodes in the tree *do* represent points at which ancestral genomes shared the same bias pattern.

8.3 Tree comparison

These results may indicate that the evolution of codon bias does not produce patterns of change recognizable with conventional genealogical methods. Alternately, it may be the case that the 64-position vectors used as input to the maximum-likelihood method simply do not have enough variation to distinguish 831 genomes meaningfully. This is more likely

because there is a great deal of dependence between all values in a synonym group, meaning that each vector does not have 64 truly independent positions. Which of these explanations (unrecognizable patterns v. insufficient resolution in the data) truly capture why maximum-likelihood tree construction fails is not possible to resolve given the available information. However, it is interesting to consider the differences between the data traditionally used to construct canonical 16S trees.

8.3.1 Tree-construction data characteristics

The canonical 16S tree is built using 16S (or even 23S) rRNA trees. These are alignments of ribosomal RNA molecules well over 1200 base-pairs in length. This size alignment – combined with a well-understood and detailed model of what differences in sequence mean for building a tree – means that the data is more then sufficient for resolving evolutionary differences between organisms, even given a large set of hundreds or thousands of sequences.

On the other hand, codon bias data is a 64-position real vector with values constrained between 0 and 1, and a rigid covariance structure between some of the columns. Furthermore, the evolutionary model being applied in a maximum-likelihood model is much more general – it is a continuous-character model originally intended for application to outward, phenotypic traits. This relative lack of data means that it is quite likely there is simply not enough information to distinguish a set of hundreds of organisms. It is an open question – beyond the scope of this study, but of possible interest for future work – if a more nuanced and specific evolutionary model of bias data would extract more tree-building power from this

data set.

8.3.2 Tree analysis

One analysis run on the tree is a comparison of adjacent leaf nodes. If a given node has the same neighbors on both trees, it would be a good indication that some shared topology is present; conversely, if most nodes do not share neighbors between the trees, it would indicate that topology is not shared. It is the latter case that is found: most leaf nodes have zero or one neighbor in common between the trees, with only a few having two and none having more. This indicates that the strongest evolutionary conclusion that can be drawn by comparing these trees is to confirm that two organisms with very recent evolutionary divergence have similar codon biases.

8.3.3 Topology comparisons

Detailed topology comparison with tree-analysis software was originally proposed (see 7). However, the software packages originally proposed for this purpose (`metatree` and `cousins`) were not designed to handle tress with hundreds of nodes, and neither package proved usable for the intended purpose. Instead, a tree scaling analysis was performed, using the tool provided by TORNADO (Sipos et al., 2010).[1] This analysis provides a quantitative and visual analysis of tree topology by comparing node branch size (the length of the branch leading to a node) and the cumulative branch size for the same node (the length of all branches in the node's subtree). Visual results of the scaling analysis are provided for the

[1] http://tornado.igb.illinois.edu/scaling.html

16S tree (fig. 8.3) and the bias tree (fig. 8.4) below. These results indicate that the two trees do indeed have substantially different topologies.[2]

8.3.4 Node neighbor analysis

In a comparison of nearest neighbors on the trees, we find that 26% of leaf nodes have no common neighbors in both trees, 74% have one neighbor in common, and there are no nodes with more then one common neighbor. This supports what has already been mentioned: that there is very little common topology between the trees. Furthermore, it also suggests that while organisms with recent common ancestry *often* have similar codon bias, they do not always – otherwise we would expect that every organism would have at least one neighbor on both trees, and that neighbor would be its most recent common ancestor.

8.4 Conclusion

The answers to the research questions addressed through phylogenetics – can we build a tree of bias evolution using common phylogenetic methods, and does any tree built using the bias data look like the canonical 16S tree – seem conclusively to be "no." This result is quite interesting: it tells us that the genealogical relationships between organisms are *not* reflected in their codon biases. The strongest evolutionary commonality appears to be that very close relatives on the 16S tree are often neighbors on the bias tree, which is explainable simply by recent common ancestry and insufficient time for their genomes to move very far apart in

[2]The same scaling analysis was conducted using the reference tree from ARB (Pruesse et al., 2007), with the same results.

bias space. Furthermore, this supports the hypothesis that codon bias provides organisms with an informational adaptive mechanism that is able to change without affecting the code of the genetic machinery: since there is no common evolutionary signal between the 16S and the bias data, we must conclude that they do not share evolutionary influences and histories.

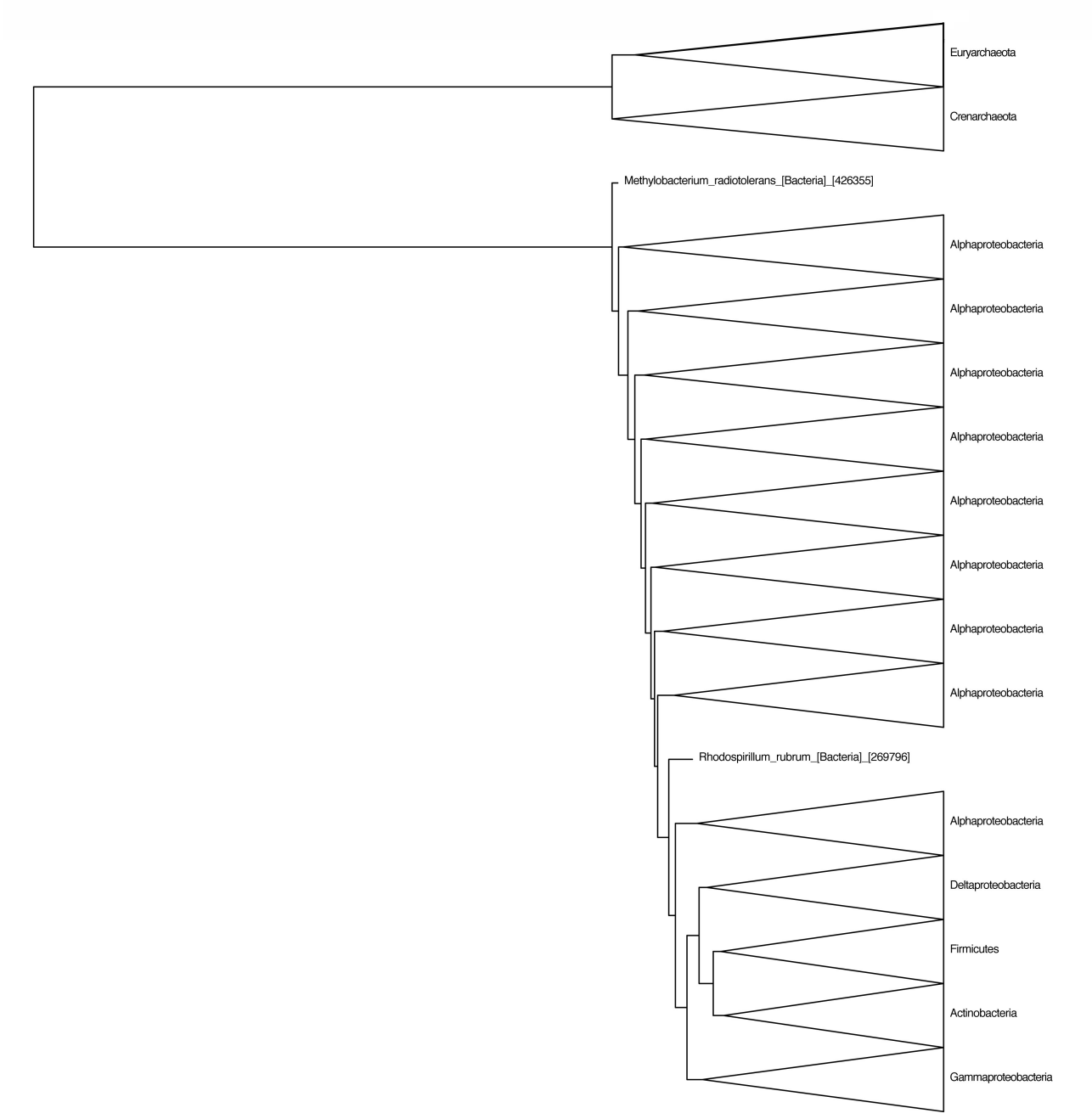

Figure 8.1: 16S rRNA tree (collapsed) constructed using `dnaml`. Major clusters on the tree collapsed for readability, and to present the general tree topology. The full tree may be seen in Appendix fig. A.1 (some magnification may be required).

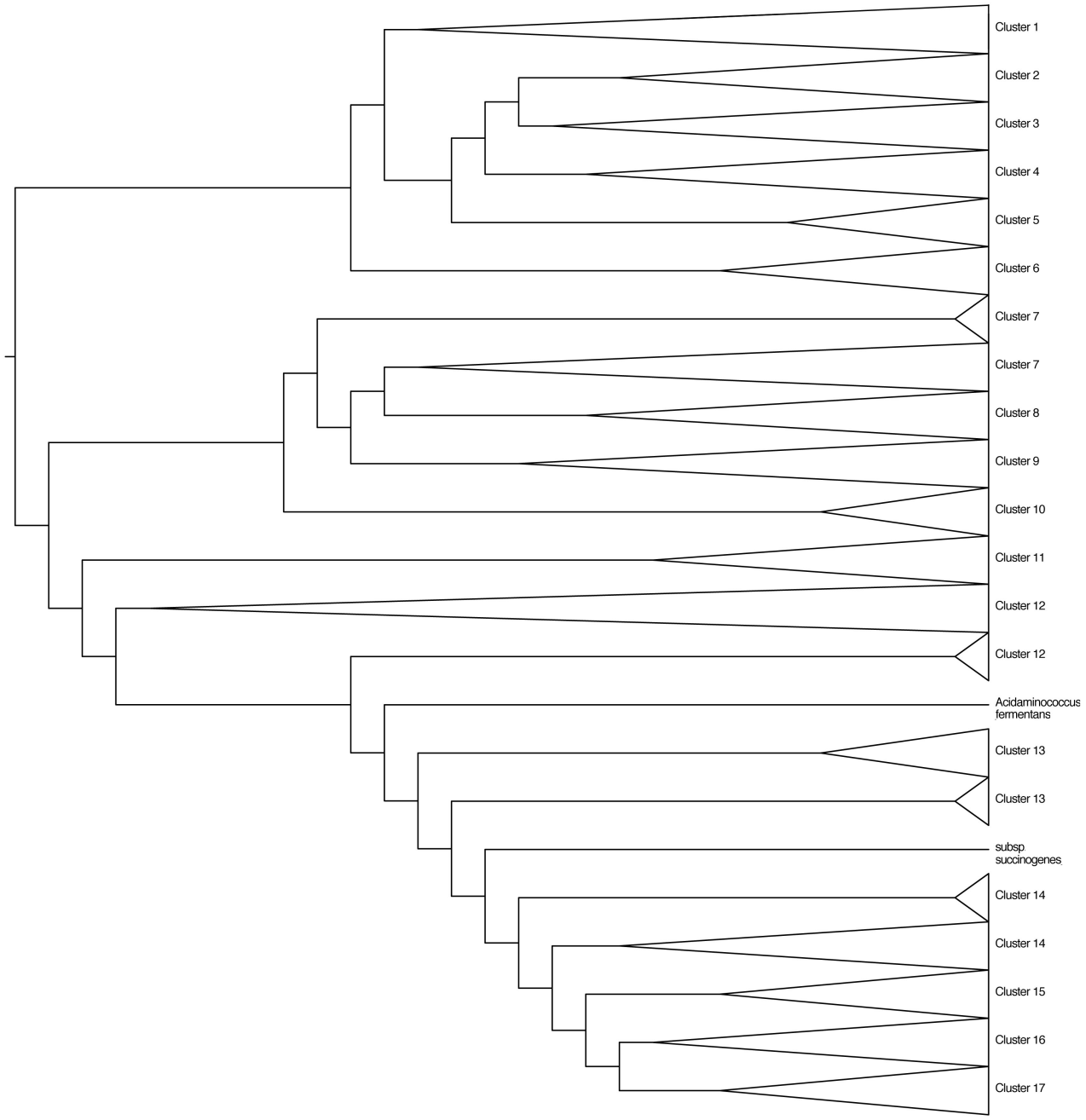

Figure 8.2: UPGMA tree constructed using a Euclidian distance measure between codon bias of genomes. Major clusters on the tree collapsed for readability, and to present the general tree topology. The full tree may be seen in fig. A.2 (some magnification may be required).

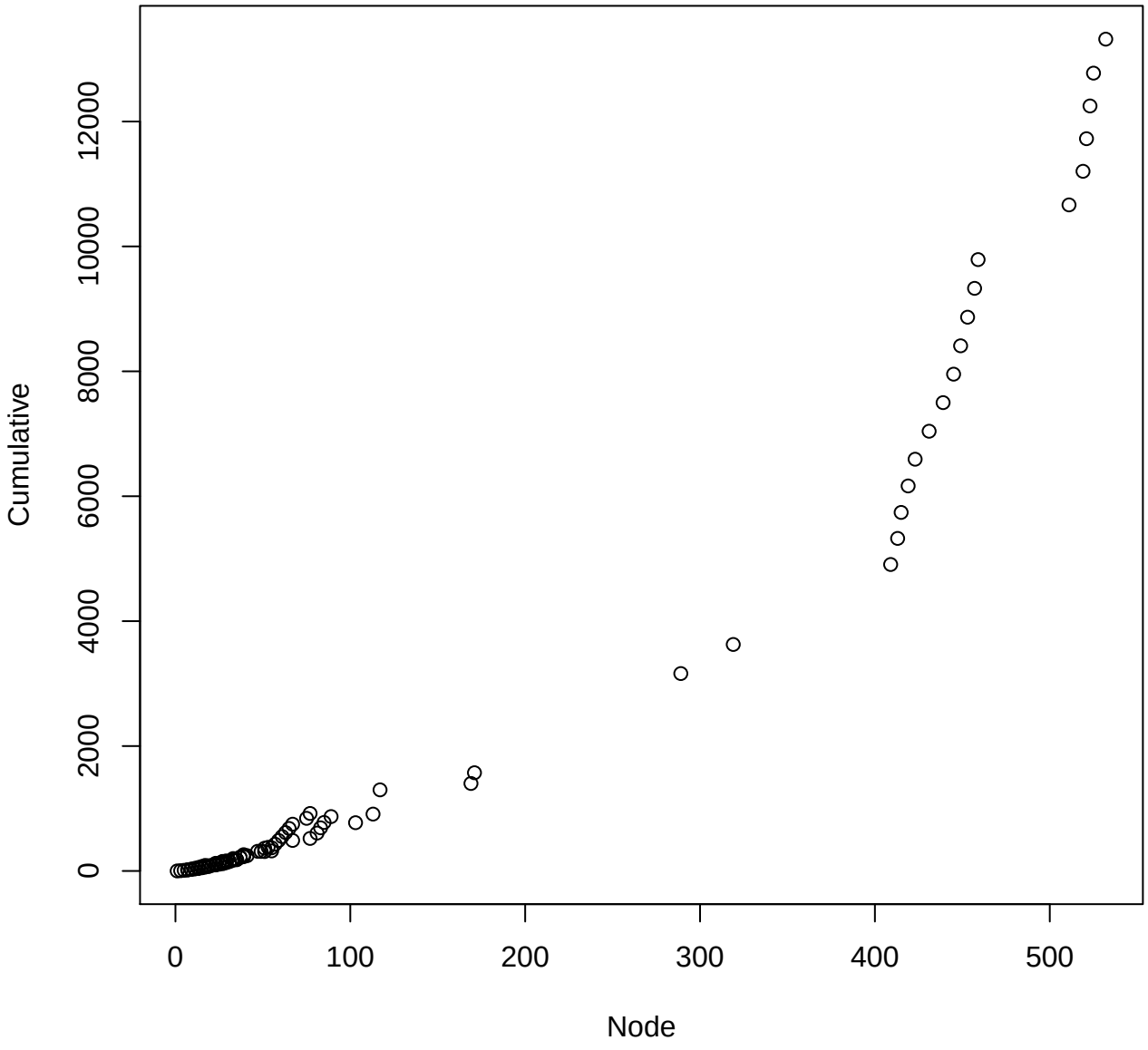

Figure 8.3: Scaling of 16S rRNA tree

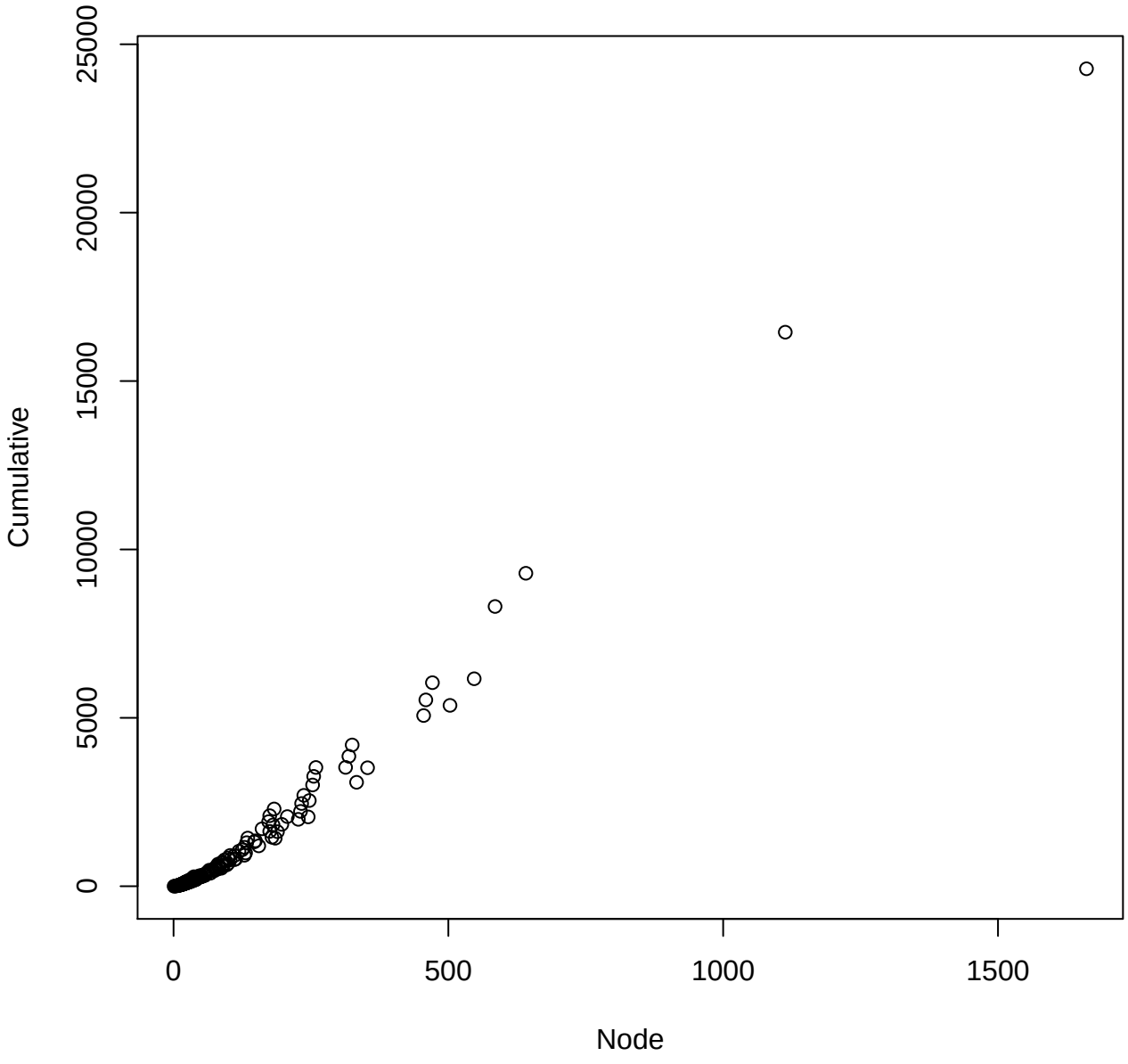

Figure 8.4: Scaling of codon bias tree

Chapter 9

Evolutionary implications and future work

The work presented here demonstrates that the long-standing core of the genetic system – the 16S rRNA – and the short-term dynamic edges of the code – codon bias – follow different evolutionary trajectories. This suggests that these components of the genetic system serve different roles in the adaptive landscape, and that understanding codon bias may reveal previously unknown aspects of the genetic code's evolutionary dynamic.

9.1 Evolutionary implications of phylogenetic results

The trees of codon bias evolution and of 16S ribosomal RNA evolution have very little topology in common (see §8.3.2 and figs. 8.3-8.4). In addition, the bias tree cannot be built using a traditional phylogenetic maximum-likelihood (ML) approach – the data does not contain enough distinguishing features to for ML to build a plausible tree. Rather, the bias tree is constructed using a distance measure in bias space and a UPGMA clustering method. The most shared feature between the trees is simply a nearest-neighbor commonality: closest relatives on the 16S tree are likely to be close together on the bias tree as well. This commonality is reflective of the recent divergence of nearest neighbors, not of some shared evolutionary dynamic or influence.

9.1.1 Possible evolutionary scenarios

There are several speculative scenarios of evolution that would lead to the sort of evolutionary patterns seen in the bias tree. I lay them out here as possibilities, and all of them would require investigation to be supported; but they make for interesting consideration, and should point in the direction of future work.

First, similar codon bias configurations could be the result of convergent evolution – that is, different organisms could arrive at the same bias because of shared evolutionary pressures. These pressures could be environmental (high/low temperature or restricted nutrient availability, for example) or competitive (living anywhere high rates of reproduction are required for a lineage to survive), or some other category of selective pressure. In order for these kinds of pressures to result in convergent evolution, it would have to be the case that particular configurations of bias space are better-suited to different pressures – rather then that *many* bias configurations are equally optimal with regard to a given factor. There is some evidence that this may be the case, at least for certain environmental factors – in particular, for thermophilic and hyperthermophilic organisms. It stands to reason that organisms in these environments might have similar adaptive codon bias, since DNA with a higher G+C content is more chemically stable at higher temperatures. Additionally, Carbone et al. (2004) has found evidence that thermophiles do indeed cluster together in bias space, though their work should be expanded to see if these results hold for organisms sequenced since they conducted their study.

Next, it may be the case that bias space is occupied as it is due to internal constraints;

that is, there are features of the genetic code and genetic machinery that mean codon bias space must look as it does under existing circumstances. We already know about some constraint structure affecting bias space – specifically, that all bias values must fall within ranges dictated by the structure of the code (see §3.1.1 for a thorough discussion). Beyond this, it may be the case that additional constraints arise from chemical interactions with the environment or other cellular components sufficient to force bias space to be structured as it is. In this scenario, there would be a direct and highly determined link between material circumstances surrounding the genetic machinery and the codon bias of an organism. While I consider this unlikely, it is a possibility, and it may be productive to investigate these hypothetical constraints – especially if other avenues of investigation do not prove fruitful.

Finally, we could imagine that codon space is occupied as it is due to random processes. This hypothesis states that there is no underlying driver for codon bias, and that the bias of a particular lineage basically wanders around in bias space – similar to the "neutral evolution" hypothesis (Kimura, 1983). The neutral evolution theory states that single-base changes to DNA sequences happen all the time, but that most of them have absolutely no effect on fitness, rendering them evolutionarily neutral – that is, they have no adaptive function. This suggests that the majority of changes in codon bias occur due to random processes, and also have no evolutionary significance. Changes in codon bias *could* be significant if they have effects on (for example) translation rate, in which case they would tend to get locked in, but the underlying process driving those changes has no structure or significance in this scenario. This then suggests that bias space has the apparent structure it does more

or less by accident; however, if this were the case I would expect a more even distribution of organisms in bias space. In any case, this final scenario should serve as a reasonable null hypothesis in future investigations.

9.1.2 Significance of tree topology

The most important conclusion suggested by the very different topologies of the 16S and bias data phylogenetic trees is that the evolutionary histories and dynamics of codon bias and the core genetic machinery have little in common. This is important for several reasons; foremost among these is the insight it provides into the nature of the genetic system and the variety of adaptive mechanism available to modern organisms.

The differing trees also reflect an important limitation in the data: the 16S is a very long (>1200 base-pair) gene sequence, while the bias data is a 64-position real vector subject to strong constraints on possible values. What this means is that the 16S data captures a great deal more evolutionary information then the bias data, which is demonstrated clearly in the difficulty in creating bias trees using more traditional phylogenetic methods. In essence, bias data is not sufficient to distinguish strongly amongst 831 genomes using a maximum-likelihood approach. This being the case, we would not expect the trees to be identical even under the best circumstances; however, they are so very unlike that I believe the conclusions described above are valid.

9.2 The big picture: codon bias as a mechanism of adaptation

So, then, what role does codon bias play in evolution, and what lessons does this system have for our understanding of information systems generally? First, codon bias adds a measure of flexibility to the genetic coding system, allowing it to adapt to changing requirements without altering the structure of the code itself. More importantly, codon bias creates a mechanism for certain kinds of adaptation (like translation rate regulation) through manipulation of the information system, *without changing the information being encoded* – it is this decoupling of the physical substrate from the information itself that creates the exhibited flexibility, and is a strong argument for the truly informational nature of the genetic system.

The role and behavior of codon bias in adaptation and evolution highlights the essential nature of information in the genetic system. Codon bias provides an effective adaptive mechanism *because* genes are primarily an informational medium. The input to the translation machinery can be changed – any codon can be substituted for a synonym – without affecting the output from the translation system. This property must be understood as informational: invariance of the message regardless of variance in the message carrier is a hallmark of other (non-biological) systems we consider informational, and it is the key to the genetic code and codon bias operating the way that it does.

9.2.1 A low-overhead adaptive mechanism

The biased encoding of genes produces what I am referring to here as a "low-overhead adaptive mechanism": that is, a mechanism that does not require changes to any significant cellular systems in order to produce an adaptive effect. Another feature of this kind of adaptive mechanism is that it is low-risk from the standpoint of lineage survival: altering codon bias is much less likely to result in an uncompetitive lineage then alteration to the genetic code itself. This is critical to the role that codon bias plays in an organism – altering things like rates of protein translation might be higher effort due to requiring external regulatory mechanisms,[1] or more risky due to the possibility of breaking translation of a particular protein entirely. The role codon bias plays is to facilitate adaptation of the protein translation process to particular circumstances through fine-tuning, rather then through gross changes to the genetic apparatus.

9.3 Future work

The present work lays out the "what" of biased coding in genomes, and makes some attempt to address the "why". Future work in this area should focus on refining and enriching our answers to why codon codon bias looks as it does, and on addressing the "how": what are the processes and evolutionary dynamics creating biased coding in the first place, and what are the mechanisms of how it changes over time?

[1] Which do, in fact, exist for more complex control scenarios – see, for example, the *lac* operon.

9.3.1 Convergent evolution

One way to address the question of why biased encoding exists as it does is to look at possible external forces – that is, outside the structure of the code itself – that constrain or shape synonymous codon use. Convergent evolution is one possible explanation for codon bias space being shaped the way it is – that is, organisms living in similar environments are driven by adaptive pressures to similar codon biases. This is a hypothesis that has been explored somewhat by Carbone et al. (2004), but a more thorough and more detailed study would be helpful. Carbone et al. found that there is evidence for some evolutionary convergence in bias space for thermophiles by looking at a set of ninety-six organisms. I would like to expand this analysis substantially, taking into account all 831 organisms in the present study and looking for correlations to many more environmental niches. This would be approached by dividing this large set of genomes up into their respective environmental niches, and looking for correlations – to support the convergent evolution hypothesis, one would expect that the codon biases within each niche would be more similar then to any biases outside the niche.

9.3.2 Evolutionary dynamics of an artificial genetic-like coding system

Another future direction for understanding the evolutionary dynamics of codon bias is to simulate a population of agents with a genetic coding-like system needing to produce "proteins" to engage in some task – probably cooperative or competitive use of resources – and

follow how codon biases change in response to a changing agent environment. The simulation environment would need to be something fairly sophisticated; to have some confidence that the dynamics being simulated had a significant resemblance to real codon bias dynamics, the system would need to capture the effect of varying biases changing translation speed, mutations affecting the bias of individual genes, and so on. Some significant work would need to be done to capture the variable features of the genetic code and codon bias in the simulation. Starting with a model of genetic code evolution like that used by Vetsigian and Goldenfeld (2009), and expanding it to include more detailed descriptions of the bias system, would be a reasonable approach.

This work would provide insight into the detailed evolutionary dynamics that lead to changing codon biases. We know from the empirical data that biases do change and that they are distributed in a particular way, and simulation work would help fill in the picture by providing an experimental system – much easier to work with than laboratory organisms – for investigation of *how* biases come to be as they are.

9.3.3 STOP codons and domain signatures

Looking at the distribution and use of the STOP codons in isolation from the rest of the codon bias data may also provide insight into the question of why bias is distributed as it is. This is because the distribution patterns of STOP codons – the synonym group that does not represent an amino acid, but acts as a control signal – should have different characteristics from the distributions of other codons. More specifically, the use of STOP codons in the

three primary domains (Archaea, Bacteria, and Eucarya) should form a domain signature pattern, similar those found at certain cites in the ribosomal RNA sequences (Roberts et al., 2008). In other words, due to the unique role of the STOP codons, it should be possible to distinguish which domain a particular organism belongs to through analysis of the STOP codons. This is because STOP is a control signal interpreted by the translation machinery, which also has domain-specific features. Furthermore, the STOP signal is not subject to the sort of environmental-adaptation pressures the other codons are since it interacts only with the rest of the translation/transcription machinery, and should therefore be influenced primarily by variations in the ribosome and other translational components.

Testing this hypothesis would require partitioning the bias data assembled for this study into STOP and non-STOP codon data, clustering the STOP data into three clusters, and checking each organism in each cluster for domain membership. If each cluster is composed solely (or at least predominately) of organisms from one domain, the hypothesis would be strongly supported.

9.4 More on heredity: some speculations for future consideration

Here are some speculations about the role of information and the fundamental nature of heredity. These ideas will require a good deal of future work, but hold promise for helping to explain some fundamental puzzles about heredity and the continuity of information through

time.

First, heredity only functions meaningfully through time – speaking of heredity as a time-free or single-instant phenomenon would be an oxymoron. On careful reflection, it is clear that this is true of all information in general – instantaneous information is not a meaningful notion. This is because information always needs a source and a sink, a sender and a recipient (Dretske, 1999) – and sending and receiving are *processes*, and therefore exist only in time. It is in this sense also that information cannot be static – though it can certainly rest for a long time between sending a receiving (or encoding and decoding), and thus *appear* static due to idleness.

What, then, is heredity's function through time? We suggest (Gasser, 2010, personal communication) that it is to maintain a population-level machinery of thermodynamic flow, even while individual elements in the population are constantly changing, losing or gaining efficiency, reproducing, and dying. This is an intuitive understanding of the problems of friction and chemical entropy: in order to have a large-scale energy-dissipating machine, one would either need to build one non-wearing and indefinitely long-lived machine (a "perpetual motion" machine, which we know to be impossible), or set up a system wherein the machinery for energy dissipation constantly renews itself, so that the mass behavior of the system always continues. It is through information that constancy – "sameness" – of the machinery through time is achieved. Our analysis is similar to that of (Smith, 2008), who approaches the question of the maintenance of organization (and information) from a perspective of physical theory. In particular, we believe that information is the *specific sort* of organization

vital to this thermodynamic process.

One example of this at the level of biological community organization was described by (Fernandez et al., 2000). They found that the overall input and output of materials from a bioreactor composed of a diverse community of microorganisms stayed more or less constant, even as the actual population levels of each species in the community changed rapidly and drastically. What this demonstrates is the ability of a biological community to organize in a way that maintains maximum resource use in an environment, despite the continuous turnover of physical materials and individual organisms.

Chapter 10

References

Avery, O. T., MacLeod, C. M., and McCarty, M. (1944). Studies on the chemical nature of the substance inducing transformation of pneumococcal types. *Journal of Experimental Biology and Medicine*, 79:137–158.

Barbieri, M. (2002). *The organic codes: an introduction to semantic biology*. Cambridge University Press, Cambridge, United Kingdom.

Bates, M. (2006). Fundamental forms of information. *Journal of the American Society of Information Science and Technology*, 57(8):1033–1045.

Beja, O., Aravind, L., Koonin, E. V., Suzuki, M. T., Hadd, A., Nguyen, L. P., Jovanovich, S. B., Gates, C. M., Feldman, R. A., Spudich, J. L., Spudich, E. N., and DeLong, E. F. (2000). Bacterial rhodopsin: Evidence for a new type of phototrophy in the sea. *Science*, 289(5486):1902–1906.

Buckland, M. (1991). Information as thing. *Journal of the American Society of Information Science*, 42(5):351–360.

Carbone, A., Képès, F., , and Zinovyev, A. (2004). Codon bias signatures, organization of microorganisms in codon space, and lifestyle. *MBE*, 23(3):547–561.

Carlson, E. A. (1991). Defining the gene: an evolving concept. *American Journal of Human Genetics*, 49:475–487.

Cooper, G. M. (2000). *The Cell*. Sinauer Associates.

Crick, F. H. C. (1968). The origin of the genetic code. *Journal of Molecular Biology*, 38:367–379.

DeLong, E. F., Preston, C. M., Mincer, T., Rich, V., Hallam, S. J., Frigaard, N.-U., Martinez, A., Sullivan, M. B., Edwards, R., Brito, B. R., Chisholm, S. W., and Karl, D. M. (2006). Community genomics among stratified microbial assemblages in the ocean's interior. *Science*, 311(5760):496–503.

Doolittle, W. F. (1999). Phylogenetic classification and the universal tree. *Science*, 284(5423):2124–2128.

Doolittle, W. F. and Bapteste, E. (2007). Pattern pluralism and the tree of life hypothesis. *Proceedings of the National Academy of Sciences*, 104(7):2043–2049.

Dose, K. (1994). On the origin of biological information. *Journal of Biological Physics*, 20(1–4):181–192.

Dretske, F. I. (1999). *Knowledge and the Flow of Information*. CSLI Publications.

Ermolaeva, M. (2001). Synonymous codon usage in bacteria. *Current issues in Molecular Biology*, 3(4):91–97.

Felsenstein, J. (2004). *Inferring Phylogenies*. Sinauer Associates, Sunderland, MA.

Felsenstein, J. (2005). *PHYLIP (Phylogeny Inference Package) version 3.6*. Department of Genome Sciences, University of Washington, Seattle. Distributed by the author.

Fernandez, A. S., Hashsham, S. A., Dollhopf, S. L., Raskin, L., Glagoleva, O., Dazzo, F. B., Hickey, R. F., Criddle, C. S., and Tiedje, J. M. (2000). Flexible community structure correlates with stable community function in methanogenic bioreactor communities perturbed by glucose. *Applied and Environmental Microbiology*, 66(9):4058–4067.

Freeland, S. A. and Hurst, L. D. (1998). The genetic code is one in a million. *Journal of Molecular Evolution*, 47:238–248.

Godfrey-Smith, P. (2000). Information, arbitrariness, and selection: Comments on Maynard-Smith. *Philosophy of Science*, 67(2):202–207.

Gogarten, J. P., Doolittle, W. F., and Lawrence, J. G. (2002). Prokaryotic evolution in light of gene transfer. *Molecular Biology and Evolution*, 19(12):2226–2238.

Hausmann, C. D., Ling, J., and Ibba, M. (2007). The unnatural culture of amino acids. *Nature Methods*, 4(3):205–206.

Huang, S. (2009). Reprogramming cell fates: reconciling rarity with robustness. *BioEssays*, 31:546–560.

Jablonka, E. and Lamb, M. J. (2006). The evolution of information in the major transitions. *Journal of Theoretical Biology*, 239:236–246.

Kandler, O. and König, H. (1998). Cell wall polymers in Archaea (Archaebacteria). *Cellular and Molecular Life Sciences*, 54(4):305–308.

Kimura, M. (1983). *The Neutral Theory of Molecular Evolution*. Cambridge University Press, Cambridge.

Laland, K. N., Odling-Smee, J., and Gilbert, S. F. (2008). Evodevo and niche construction: Building bridges. *Journal of Experimental Zoology*, 310B:549–566.

Landweber, L. F., Kuo, T., and Curtis, E. A. (2000). Evolution and assembly of an extremely scrambled gene. *Proceedings of the National Academy of Sciences*, 97(7):3298–3303.

Martienssen, R. A. and Colot, V. (2001). Dna methylation and epigenetic inheritance in plants and filamentous fungi. *Science*, 293:1070–1074.

Maynard-Smith, J. (2000). The concept of information in biology. *Philosophy of Science*, 67(2):177–195.

Mendel, G. (1866). Versuche über pflanzen-hybriden. *Verh. Naturforsch. Ver. Brünn*, 4:3–47.

Nakamura, Y., Gojobori, T., and Ikemura, T. (2000). Codon usage tabulated from the international dna sequence databases: status for the year 2000. *Nucleic Acids Research*, 28:292.

Nanney, D. L. (1968). Cortical patterns in cellular morphogenesis. *Science*, 160:496–502.

Nirenberg, M. W. and Matthaei, J. H. (1961). The dependence of cell-free protein synthesis in e. coli upon naturally occurring or synthetic polynucleotides. *Proceedings of the National Academy of Sciences*, 47:1588–1594.

Nye, T. M. W. (2008). Trees of trees: An approach to comparing multiple alternative phylogenies. *Systematic Biology*, 57(5):785–794.

Ochoa, S. (1963). Synthetic polynucleotides and the genetic code. *Federation Proceedings*, 22:62–74.

Pruesse, E., Quast, C., Knittel, K., Fuchs, B. M., Ludwig, W., Peplies, J., and Glöckner, F. O. (2007). Silva: a comprehensive online resource for quality checked and aligned ribosomal rna sequence data compatible with arb. *Nucleic Acids Research*, 35(21):7188–7196.

Qin, H., Wu, W. B., Comeron, J. M., Kreitman, M., and Li, W.-H. (2004). Intragentic spatial patterns of codon usage bias in prokaryotic and eukaryotic genomes. *Genetics*, 168:2245–2260.

Roberts, E., Sethi, A., Montoya, J., Woese, C. R., and Luthey-Schulten, Z. (2008). Molecular signatures of ribosomal evolution. *Proceedings of the National Academy of Sciences*, 105(37):13953–13958.

Sarkar, S. (2000). Information in genetics and developmental biology: Comments on Maynard-Smith. *Philosophy of Science*, 67(2):208–213.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423 and 623–656.

Short, J. E., Bohn, R. E., and Baru, C. (2010). How much information? 2010 report on enterprise server information. Technical report, University of California at San Diego.

Sipos, M., Jeraldo, P., Chia, N., Qu, A., Dhillon, A. S., Konkel, M. E., Nelson, K. E., White, B. A., and Goldenfeld, N. (2010). Robust computational analysis of rrna hypervariable tag datasets. *PLoS ONE*, 5(12):e15220.

Smith, E. (2008). Thermodynamcs of natural selection i: Energy flow and the limits on organization. *Journal of Theoretical Biology*, 252:185–197.

Sterelny, K. (2000). The "genetic program" program: A commentary on Maynard-Smith on information in biology. *Philosophy of Science*, 67(2):195–201.

Vetsigian, K. and Goldenfeld, N. (2009). Genome rhetoric and the emergence of compositional bias. *Proceedings of the National Academy of Sciences*, 106(1):215–220.

Vetsigian, K., Woese, C., and Goldenfeld, N. (2006). Collective evolution and the genetic code. *Proceedings of the National Academy of Sciences*, 103(28):10696–10701.

Wang, Q., Parrish, A. R., and Wang, L. (2009). Expanding the genetic code for biological studies. *Chemistry & Biology*, 16:323–336.

Watson, J. D. and Crick, F. H. C. (1953a). Genetical implications of the structure of deoxyribonucleic acid. *Nature*, 171:964.

Watson, J. D. and Crick, F. H. C. (1953b). Molecular structure of nucleic acids. *Nature*, 171:737–738.

Wheelis, M. L., Kandler, O., and Woese, C. R. (1992). On the nature of global classification. *Proceedings of the National Academy of Sciences of the United States of America*, 89(7):2930–2934.

Woese, C. R. (2002). On the evolution of cells. *Proceedings of the National Academy of Sciences*, 99(13):8742–8747.

Woese, C. R., Dugre, D. H., Saxinger, W. C., and Dugre, S. A. (1966). The molecular basis for the genetic code. *Proceedings of the National Academy of Sciences*, 55(4):966–974.

Woese, C. R. and Fox, G. E. (1978). Phylogenetic structure of the prokaryotic domain: the primary kingdoms. *Proceedings of the National Academy of Sciences*, 74(11):5088–5090.

Woese, C. R., Olsen, G. J., Ibba, M., and Söll, D. (2000). Aminoacyl-tRNA synthetases, the genetic code, and the evolutionary process. *Microbiological and Molecular Biological Reviews*, 64(1):202–236.

Appendix A

Full phylogenetic trees

Full phylogenetic trees. Some magnification of this PDF document may be required to read node names and resolve tree branching.

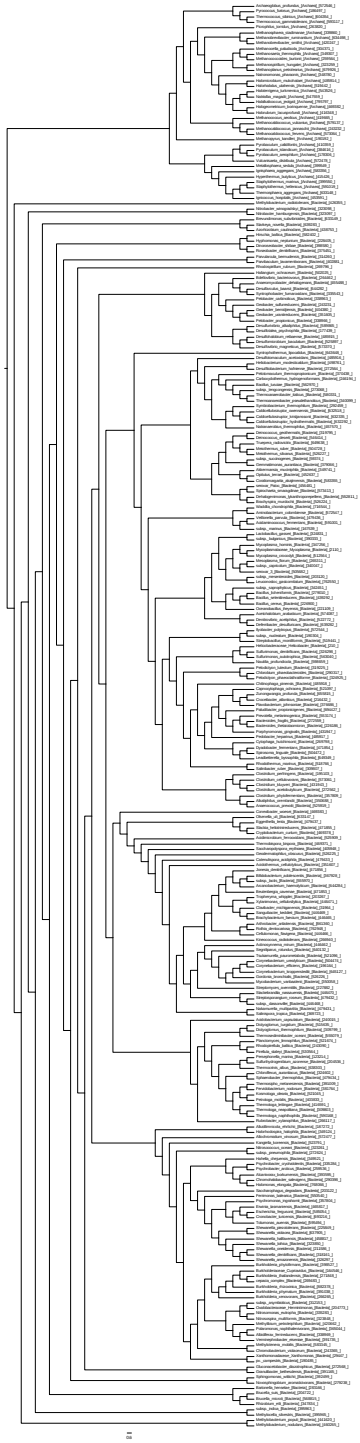

Figure A.1: 16S rRNA tree constructed using `dnaml`

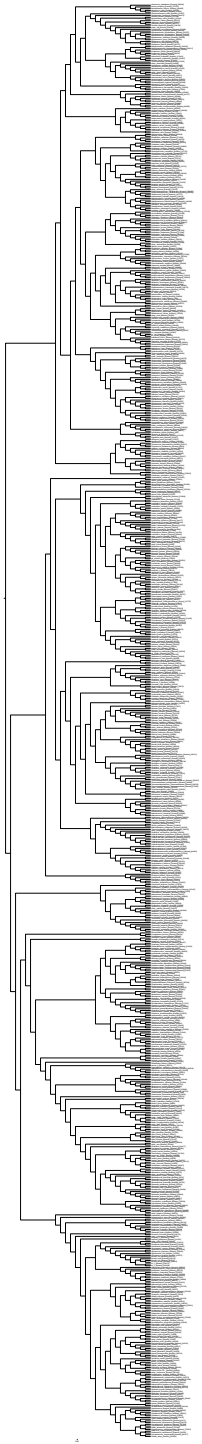

Figure A.2: UPGMA tree constructed using a Euclidian distance measure between codon bias of genomes

Appendix B

Organisms in bias analysis

Table B.1: Full list or organisms in bias analysis.

Organism	**Domain**	**TaxID**
Acaryochloris marina	Bacteria	329726
Accumulibacter phosphatis	Bacteria	522306
Acetobacter pasteurianus	Bacteria	634452
Acetohalobium arabaticum	Bacteria	574087
Acholeplasma laidlawii	Bacteria	441768
Achromobacter xylosoxidans	Bacteria	762376
Acidaminococcus fermentans	Bacteria	591001
Acidilobus saccharovorans	Archaea	666510
Acidimicrobium ferrooxidans	Bacteria	525909
Acidiphilium cryptum	Bacteria	349163
Acidithiobacillus ferrooxidans	Bacteria	243159
Acidobacterium capsulatum	Bacteria	240015
Acidothermus cellulolyticus	Bacteria	351607
Acidovorax citrulli	Bacteria	397945
Acidovorax ebreus	Bacteria	535289
Aciduliprofundum boonei	Archaea	439481
Acinetobacter baumannii	Bacteria	480119
Actinobacillus succinogenes	Bacteria	339671
Actinosynnema mirum	Bacteria	446462
Acyrthosiphon pisum	Bacteria	563178
Aeropyrum pernix	Archaea	272557
Aggregatibacter actinomycetemcomitans	Bacteria	668336
Aggregatibacter aphrophilus	Bacteria	634176
Agrobacterium tumefaciens	Bacteria	176299
Agrobacterium tumefaciens	Bacteria	311403
Agrobacterium vitis	Bacteria	311402
Akkermansia muciniphila	Bacteria	349741
Albidiferax ferrireducens	Bacteria	338969
Alcanivorax borkumensis	Bacteria	393595
Aliivibrio fischeri	Bacteria	312309
Aliivibrio salmonicida	Bacteria	316275
Alkalilimnicola ehrlichii	Bacteria	187272
Alkaliphilus metalliredigens	Bacteria	293826
Alkaliphilus oremlandii	Bacteria	350688
Allochromatium vinosum	Bacteria	572477
Alteromonas macleodii	Bacteria	314275
Aminobacterium colombiense	Bacteria	572547
Ammonifex degensii	Bacteria	429009
Amoebophilus asiaticus	Bacteria	452471
Amycolatopsis mediterranei	Bacteria	749927
Anabaena azollae	Bacteria	551115

Organism	Domain	TaxID
Anabaena variabilis	Bacteria	240292
Anaerococcus prevotii	Bacteria	525919
Anaeromyxobacter dehalogenans	Bacteria	455488
Anaplasma centrale	Bacteria	574556
Anaplasma marginale	Bacteria	320483
Anaplasma phagocytophilum	Bacteria	212042
Anoxybacillus flavithermus	Bacteria	491915
Aquifex aeolicus	Bacteria	224324
Aquificaceae Hydrogenobaculum	Bacteria	380749
Arcanobacterium haemolyticum	Bacteria	644284
Archaeoglobus fulgidus	Archaea	224325
Archaeoglobus profundus	Archaea	572546
Arcobacter butzleri	Bacteria	367737
Arcobacter nitrofigilis	Bacteria	572480
Aromatoleum aromaticum	Bacteria	76114
Arthrobacter arilaitensis	Bacteria	861360
Arthrobacter aurescens	Bacteria	290340
Arthrobacter chlorophenolicus	Bacteria	452863
Atopobium parvulum	Bacteria	521095
Azobacteroides pseudotrichonymphae	Bacteria	511995
Azorhizobium caulinodans	Bacteria	438753
Azotobacter vinelandii	Bacteria	322710
Bacillaceae Geobacillus	Bacteria	471223
Bacillaceae Geobacillus	Bacteria	544556
Bacillaceae Geobacillus	Bacteria	581103
Bacillaceae Geobacillus	Bacteria	691437
Bacillus amyloliquefaciens	Bacteria	692420
Bacillus anthracis	Bacteria	261594
Bacillus atrophaeus	Bacteria	720555
Bacillus cereus	Bacteria	572264
Bacillus clausii	Bacteria	66692
Bacillus cytotoxicus	Bacteria	315749
Bacillus halodurans	Bacteria	272558
Bacillus licheniformis	Bacteria	279010
Bacillus megaterium	Bacteria	592022
Bacillus pseudofirmus	Bacteria	398511
Bacillus pumilus	Bacteria	315750
Bacillus selenitireducens	Bacteria	439292
Bacillus thuringiensis	Bacteria	412694
Bacillus tusciae	Bacteria	562970
Bacillus weihenstephanensis	Bacteria	315730

Organism	Domain	TaxID
Bacteroides fragilis	Bacteria	272559
Bacteroides thetaiotaomicron	Bacteria	226186
Bacteroides vulgatus	Bacteria	435590
Bartonella bacilliformis	Bacteria	360095
Bartonella grahamii	Bacteria	634504
Bartonella henselae	Bacteria	283166
Bartonella quintana	Bacteria	283165
Bartonella tribocorum	Bacteria	382640
Baumannia cicadellinicola	Bacteria	374463
Bdellovibrio bacteriovorus	Bacteria	264462
Beutenbergia cavernae	Bacteria	471853
Bifidobacterium adolescentis	Bacteria	367928
Bifidobacterium bifidum	Bacteria	702459
Bifidobacterium dentium	Bacteria	401473
Blattella germanica	Bacteria	331104
Bordetella avium	Bacteria	360910
Bordetella bronchiseptica	Bacteria	257310
Bordetella parapertussis	Bacteria	257311
Bordetella pertussis	Bacteria	257313
Bordetella petrii	Bacteria	340100
Borrelia afzelii	Bacteria	390236
Borrelia bavariensis	Bacteria	290434
Borrelia burgdorferi	Bacteria	224326
Borrelia duttonii	Bacteria	412419
Borrelia hermsii	Bacteria	314723
Borrelia recurrentis	Bacteria	412418
Borrelia turicatae	Bacteria	314724
botulinum A	Bacteria	413999
Brachybacterium faecium	Bacteria	446465
Brachyspira hyodysenteriae	Bacteria	565034
Brachyspira murdochii	Bacteria	526224
Brachyspira pilosicoli	Bacteria	759914
Bradyrhizobiaceae Bradyrhizobium	Bacteria	114615
Bradyrhizobiaceae Bradyrhizobium	Bacteria	288000
Bradyrhizobium japonicum	Bacteria	224911
Brevibacillus brevis	Bacteria	358681
Brevundimonas subvibrioides	Bacteria	633149
Brucella canis	Bacteria	483179
Brucella microti	Bacteria	568815
Brucella ovis	Bacteria	444178
Brucella suis	Bacteria	204722

Organism	Domain	TaxID
Burkholderia ambifaria	Bacteria	339670
Burkholderiaceae Burkholderia	Bacteria	640511
Burkholderiaceae Burkholderia	Bacteria	640512
Burkholderiaceae Cupriavidus	Bacteria	164546
Burkholderia cenocepacia	Bacteria	331271
Burkholderia glumae	Bacteria	626418
Burkholderia mallei	Bacteria	243160
Burkholderia multivorans	Bacteria	395019
Burkholderia phymatum	Bacteria	391038
Burkholderia phytofirmans	Bacteria	398527
Burkholderia pseudomallei	Bacteria	357348
Burkholderia rhizoxinica	Bacteria	882378
Burkholderia thailandensis	Bacteria	271848
Burkholderia vietnamiensis	Bacteria	269482
Burkholderia xenovorans	Bacteria	266265
bv. 1	Bacteria	262698
bv. 2	Bacteria	546272
bv. trifolii	Bacteria	395491
Caldicellulosiruptor bescii	Bacteria	521460
Caldicellulosiruptor hydrothermalis	Bacteria	632292
Caldicellulosiruptor kristjanssonii	Bacteria	632335
Caldicellulosiruptor kronotskyensis	Bacteria	632348
Caldicellulosiruptor obsidiansis	Bacteria	608506
Caldicellulosiruptor owensensis	Bacteria	632518
Caldicellulosiruptor saccharolyticus	Bacteria	351627
Caldivirga maquilingensis	Archaea	397948
Campylobacter concisus	Bacteria	360104
Campylobacter curvus	Bacteria	360105
Campylobacter hominis	Bacteria	360107
Campylobacter lari	Bacteria	306263
Candidatus Blochmannia	Bacteria	203907
Capnocytophaga ochracea	Bacteria	521097
Carboxydothermus hydrogenoformans	Bacteria	246194
Carsonella ruddii	Bacteria	387662
Catenulispora acidiphila	Bacteria	479433
Caulobacteraceae Caulobacter	Bacteria	366602
Caulobacter segnis	Bacteria	509190
Caulobacter vibrioides	Bacteria	190650
Cellulomonas flavigena	Bacteria	446466
Cellvibrio japonicus	Bacteria	498211
cepacia complex	Bacteria	269483

Organism	Domain	TaxID
Chitinophaga pinensis	Bacteria	485918
Chlamydia muridarum	Bacteria	243161
Chlamydia trachomatis	Bacteria	471472
Chlamydophila abortus	Bacteria	218497
Chlamydophila caviae	Bacteria	227941
Chlamydophila felis	Bacteria	264202
Chlorobaculum parvum	Bacteria	517417
Chlorobaculum tepidum	Bacteria	194439
Chlorobium chlorochromatii	Bacteria	340177
Chlorobium limicola	Bacteria	290315
Chlorobium phaeobacteroides	Bacteria	331678
Chlorobium phaeovibrioides	Bacteria	290318
Chloroflexaceae Chloroflexus	Bacteria	480224
Chloroflexaceae Roseiflexus	Bacteria	357808
Chloroflexus aggregans	Bacteria	326427
Chloroflexus aurantiacus	Bacteria	324602
Chloroherpeton thalassium	Bacteria	517418
Chromobacterium violaceum	Bacteria	243365
Chromohalobacter salexigens	Bacteria	290398
Chroococcales Cyanothece	Bacteria	43989
Chroococcales Cyanothece	Bacteria	65393
Chroococcales Synechococcus	Bacteria	110662
Chroococcales Synechococcus	Bacteria	316278
Chroococcales Synechococcus	Bacteria	316279
Chroococcales Synechococcus	Bacteria	32049
Chroococcales Synechococcus	Bacteria	32051
Chroococcales Synechococcus	Bacteria	321332
Chroococcales Synechococcus	Bacteria	64471
Chroococcales Synechococcus	Bacteria	84588
Chroococcales Synechocystis	Bacteria	1148
Citrobacter koseri	Bacteria	290338
Citrobacter rodentium	Bacteria	637910
Clostridium acetobutylicum	Bacteria	272562
Clostridium beijerinckii	Bacteria	290402
Clostridium cellulolyticum	Bacteria	394503
Clostridium cellulovorans	Bacteria	573061
Clostridium difficile	Bacteria	272563
Clostridium kluyveri	Bacteria	431943
Clostridium ljungdahlii	Bacteria	748727
Clostridium novyi	Bacteria	386415
Clostridium perfringens	Bacteria	195102

Organism	Domain	TaxID
Clostridium phytofermentans	Bacteria	357809
Clostridium proteoclasticum	Bacteria	515622
Clostridium saccharolyticum	Bacteria	610130
Clostridium sticklandii	Bacteria	499177
Clostridium tetani	Bacteria	212717
Clostridium thermocellum	Bacteria	203119
Colwellia psychrerythraea	Bacteria	167879
Comamonadaceae Acidovorax	Bacteria	232721
Comamonadaceae Polaromonas	Bacteria	296591
Conexibacter woesei	Bacteria	469383
Coprothermobacter proteolyticus	Bacteria	309798
Coraliomargarita akajimensis	Bacteria	583355
Corynebacterium aurimucosum	Bacteria	548476
Corynebacterium diphtheriae	Bacteria	257309
Corynebacterium efficiens	Bacteria	196164
Corynebacterium glutamicum	Bacteria	196627
Corynebacterium jeikeium	Bacteria	306537
Corynebacterium kroppenstedtii	Bacteria	645127
Corynebacterium pseudotuberculosis	Bacteria	765874
Corynebacterium urealyticum	Bacteria	504474
Coxiella burnetii	Bacteria	434923
Croceibacter atlanticus	Bacteria	216432
Cronobacter sakazakii	Bacteria	290339
Cronobacter turicensis	Bacteria	693216
Cryptobacterium curtum	Bacteria	469378
Culex quinquefasciatus	Bacteria	570417
Cupriavidus metallidurans	Bacteria	266264
Cupriavidus necator	Bacteria	381666
Cytophaga hutchinsonii	Bacteria	269798
Dechloromonas aromatica	Bacteria	159087
Deferribacter desulfuricans	Bacteria	639282
Dehalococcoides ethenogenes	Bacteria	243164
Dehalococcoidetes Dehalococcoides	Bacteria	216389
Dehalococcoidetes Dehalococcoides	Bacteria	255470
Dehalococcoidetes Dehalococcoides	Bacteria	311424
Dehalococcoidetes Dehalococcoides	Bacteria	633145
Dehalogenimonas lykanthroporepellens	Bacteria	552811
Deinococcus deserti	Bacteria	546414
Deinococcus geothermalis	Bacteria	319795
Deinococcus radiodurans	Bacteria	243230
Delftia acidovorans	Bacteria	398578

Organism	Domain	TaxID
Denitrovibrio acetiphilus	Bacteria	522772
Desulfarculus baarsii	Bacteria	644282
Desulfatibacillum alkenivorans	Bacteria	439235
Desulfitobacterium hafniense	Bacteria	272564
Desulfobacterium autotrophicum	Bacteria	177437
Desulfococcus oleovorans	Bacteria	96561
Desulfohalobium retbaense	Bacteria	485915
Desulfomicrobium baculatum	Bacteria	525897
Desulforudis audaxviator	Bacteria	477974
Desulfotalea psychrophila	Bacteria	177439
Desulfotomaculum acetoxidans	Bacteria	485916
Desulfotomaculum reducens	Bacteria	349161
Desulfovibrio magneticus	Bacteria	573370
Desulfovibrio salexigens	Bacteria	526222
Desulfovibrio vulgaris	Bacteria	883
Desulfurivibrio alkaliphilus	Bacteria	589865
Desulfurococcus kamchatkensis	Archaea	490899
Dichelobacter nodosus	Bacteria	246195
Dickeya dadantii	Bacteria	198628
Dickeya zeae	Bacteria	561229
Dictyoglomus thermophilum	Bacteria	309799
Dictyoglomus turgidum	Bacteria	515635
Dinoroseobacter shibae	Bacteria	398580
Dyadobacter fermentans	Bacteria	471854
Ectothiorhodospiraceae Thioalkalivibrio	Bacteria	396588
Ectothiorhodospiraceae Thioalkalivibrio	Bacteria	396595
Edwardsiella ictaluri	Bacteria	634503
Edwardsiella tarda	Bacteria	498217
Eggerthella lenta	Bacteria	479437
Ehrlichia canis	Bacteria	269484
Ehrlichia chaffeensis	Bacteria	205920
Ehrlichia ruminantium	Bacteria	302409
Elusimicrobium minutum	Bacteria	445932
Enterobacteriaceae Enterobacter	Bacteria	399742
Enterobacteriaceae Photorhabdus	Bacteria	291112
Enterococcus faecalis	Bacteria	226185
environmental samples	Archaea	351160
Epsilonproteobacteria Nitratiruptor	Bacteria	387092
Epsilonproteobacteria Sulfurovum	Bacteria	387093
Erwinia amylovora	Bacteria	716540
Erwinia billingiae	Bacteria	634500

Organism	Domain	TaxID
Erwinia pyrifoliae	Bacteria	634499
Erwinia tasmaniensis	Bacteria	465817
Erythrobacter litoralis	Bacteria	314225
Escherichia coli	Bacteria	362663
Escherichia fergusonii	Bacteria	585054
Eubacterium eligens	Bacteria	515620
Eubacterium limosum	Bacteria	903814
Eubacterium rectale	Bacteria	515619
Exiguobacterium sibiricum	Bacteria	262543
Ferrimonas balearica	Bacteria	550540
Ferroglobus placidus	Archaea	589924
Fervidobacterium nodosum	Bacteria	381764
Finegoldia magna	Bacteria	334413
Flavobacteriaceae Maribacter	Bacteria	313603
Flavobacterium johnsoniae	Bacteria	376686
Flavobacterium psychrophilum	Bacteria	402612
flexneri 2a	Bacteria	198214
Francisella novicida	Bacteria	401614
Francisella philomiragia	Bacteria	484022
Frankia alni	Bacteria	326424
Frankiaceae Frankia	Bacteria	106370
Frankiaceae Frankia	Bacteria	298653
Frankiaceae Frankia	Bacteria	298654
Gallionella capsiferriformans	Bacteria	395494
Gammaproteobacteria miscellaneous	Bacteria	83406
Gardnerella vaginalis	Bacteria	553190
Gemmatimonas aurantiaca	Bacteria	379066
genomosp. BVAB3	Bacteria	699246
Geobacillus kaustophilus	Bacteria	235909
Geobacillus thermodenitrificans	Bacteria	420246
Geobacteraceae Geobacter	Bacteria	316067
Geobacteraceae Geobacter	Bacteria	443144
Geobacter bemidjiensis	Bacteria	404380
Geobacter lovleyi	Bacteria	398767
Geobacter metallireducens	Bacteria	269799
Geobacter sulfurreducens	Bacteria	243231
Geobacter uraniireducens	Bacteria	351605
Geodermatophilus obscurus	Bacteria	526225
gill symbiont	Bacteria	412965
Gloeobacter violaceus	Bacteria	251221
Gluconacetobacter diazotrophicus	Bacteria	272568

Organism	Domain	TaxID
Gluconobacter oxydans	Bacteria	290633
Gokushovirinae Chlamydiamicrovirus	viruses	117575
Gordonia bronchialis	Bacteria	526226
Gramella forsetii	Bacteria	411154
Granulibacter bethesdensis	Bacteria	391165
Haemophilus ducreyi	Bacteria	233412
Haemophilus influenzae	Bacteria	281310
Haemophilus parasuis	Bacteria	557723
Hahella chejuensis	Bacteria	349521
Halalkalicoccus jeotgali	Archaea	795797
Halanaerobiaceae Halanaerobium	Bacteria	656519
Haliangium ochraceum	Bacteria	502025
Haloarcula marismortui	Archaea	272569
Halobacterium salinarum	Archaea	478009
Halobacterium salinarum	Archaea	64091
Haloferax volcanii	Archaea	309800
Halogeometricum borinquense	Archaea	469382
Halomicrobium mukohataei	Archaea	485914
Halomonas elongata	Bacteria	768066
Haloquadratum walsbyi	Archaea	362976
Halorhabdus utahensis	Archaea	519442
Halorhodospira halophila	Bacteria	349124
Halorubrum lacusprofundi	Archaea	416348
Haloterrigena turkmenica	Archaea	543526
Halothermothrix orenii	Bacteria	373903
Halothiobacillus neapolitanus	Bacteria	555778
Hamiltonella defensa	Bacteria	572265
Helicobacter acinonychis	Bacteria	382638
Helicobacter mustelae	Bacteria	679897
Helicobacter pylori	Bacteria	85962
Heliobacterium modesticaldum	Bacteria	498761
Herbaspirillum seropedicae	Bacteria	757424
Herpetosiphon aurantiacus	Bacteria	316274
Hirschia baltica	Bacteria	582402
Histophilus somni	Bacteria	205914
Hydrogenobacter thermophilus	Bacteria	608538
Hydrogenothermaceae Sulfurihydrogenibium	Bacteria	436114
Hyperthermus butylicus	Archaea	415426
Hyphomicrobium denitrificans	Bacteria	582899
Hyphomonas neptunium	Bacteria	228405
Idiomarina loihiensis	Bacteria	283942

Organism	Domain	TaxID
Ignicoccus hospitalis	Archaea	453591
Ignisphaera aggregans	Archaea	583356
Ilyobacter polytropus	Bacteria	572544
Jonesia denitrificans	Bacteria	471856
Kangiella koreensis	Bacteria	523791
Ketogulonicigenium vulgare	Bacteria	880591
Kineococcus radiotolerans	Bacteria	266940
Klebsiella pneumoniae	Bacteria	507522
Klebsiella variicola	Bacteria	640131
Kocuria rhizophila	Bacteria	378753
Korarchaeum cryptofilum	Archaea	374847
Koribacter versatilis	Bacteria	204669
Kosmotoga olearia	Bacteria	521045
Kribbella flavida	Bacteria	479435
Kytococcus sedentarius	Bacteria	478801
Lactobacillus acidophilus	Bacteria	272621
Lactobacillus brevis	Bacteria	387344
Lactobacillus casei	Bacteria	321967
Lactobacillus crispatus	Bacteria	748671
Lactobacillus fermentum	Bacteria	334390
Lactobacillus gasseri	Bacteria	324831
Lactobacillus helveticus	Bacteria	405566
Lactobacillus johnsonii	Bacteria	633699
Lactobacillus plantarum	Bacteria	220668
Lactobacillus reuteri	Bacteria	557436
Lactobacillus rhamnosus	Bacteria	568703
Lactobacillus salivarius	Bacteria	362948
Laribacter hongkongensis	Bacteria	557598
Lawsonia intracellularis	Bacteria	363253
Leadbetterella byssophila	Bacteria	649349
Legionella longbeachae	Bacteria	661367
Leptothrix cholodnii	Bacteria	395495
Leptotrichia buccalis	Bacteria	523794
Leuconostoc citreum	Bacteria	349519
Leuconostoc gasicomitatum	Bacteria	762550
Leuconostoc kimchii	Bacteria	762051
Liberibacter asiaticus	Bacteria	537021
Listeria innocua	Bacteria	272626
Listeria monocytogenes	Bacteria	169963
Listeria seeligeri	Bacteria	683837
Listeria welshimeri	Bacteria	386043

Organism	Domain	TaxID
Lysinibacillus sphaericus	Bacteria	444177
Macrococcus caseolyticus	Bacteria	458233
Magnetospirillum magneticum	Bacteria	342108
Mannheimia succiniciproducens	Bacteria	221988
Maricaulis maris	Bacteria	394221
Marinobacter hydrocarbonoclasticus	Bacteria	351348
Meiothermus ruber	Bacteria	504728
Meiothermus silvanus	Bacteria	526227
Mesorhizobium loti	Bacteria	266835
Metallosphaera sedula	Archaea	399549
Methanobrevibacter ruminantium	Archaea	634498
Methanobrevibacter smithii	Archaea	420247
Methanocaldococcaceae Methanocaldococcus	Archaea	644281
Methanocaldococcus fervens	Archaea	573064
Methanocaldococcus infernus	Archaea	573063
Methanocaldococcus jannaschii	Archaea	243232
Methanocaldococcus vulcanius	Archaea	579137
Methanocella paludicola	Archaea	304371
Methanococcoides burtonii	Archaea	259564
Methanococcus aeolicus	Archaea	419665
Methanococcus maripaludis	Archaea	402880
Methanococcus vannielii	Archaea	406327
Methanococcus voltae	Archaea	456320
Methanocorpusculum labreanum	Archaea	410358
Methanoculleus marisnigri	Archaea	368407
Methanohalobium evestigatum	Archaea	644295
Methanohalophilus mahii	Archaea	547558
Methanoplanus petrolearius	Archaea	679926
Methanopyrus kandleri	Archaea	190192
Methanoregula boonei	Archaea	456442
Methanosaeta thermophila	Archaea	349307
Methanosarcina acetivorans	Archaea	188937
Methanosarcina barkeri	Archaea	269797
Methanosarcina mazei	Archaea	192952
Methanosphaera stadtmanae	Archaea	339860
Methanosphaerula palustris	Archaea	521011
Methanospirillum hungatei	Archaea	323259
Methanothermobacter marburgensis	Archaea	79929
Methanothermobacter thermautotrophicus	Archaea	187420
Methanothermus fervidus	Archaea	523846
Methylacidiphilum infernorum	Bacteria	481448

Organism	Domain	TaxID
Methylibium petroleiphilum	Bacteria	420662
Methylobacillus flagellatus	Bacteria	265072
Methylobacteriaceae Methylobacterium	Bacteria	426117
Methylobacterium chloromethanicum	Bacteria	440085
Methylobacterium extorquens	Bacteria	272630
Methylobacterium nodulans	Bacteria	460265
Methylobacterium populi	Bacteria	441620
Methylobacterium radiotolerans	Bacteria	426355
Methylocella silvestris	Bacteria	395965
Methylococcus capsulatus	Bacteria	243233
Methylophilaceae Methylotenera	Bacteria	666681
Methylophilaceae Methylovorus	Bacteria	582744
Methylophilaceae Methylovorus	Bacteria	887061
Methylotenera mobilis	Bacteria	583345
Micrococcaceae Arthrobacter	Bacteria	290399
Micrococcus luteus	Bacteria	465515
Microcystis aeruginosa	Bacteria	449447
Micromonospora aurantiaca	Bacteria	644283
Mobiluncus curtisii	Bacteria	548479
Moorella thermoacetica	Bacteria	264732
Moraxella catarrhalis	Bacteria	749219
Moraxellaceae Acinetobacter	Bacteria	436717
Moraxellaceae Acinetobacter	Bacteria	62977
Moraxellaceae Psychrobacter	Bacteria	349106
Mycobacteriaceae Mycobacterium	Bacteria	164756
Mycobacteriaceae Mycobacterium	Bacteria	164757
Mycobacteriaceae Mycobacterium	Bacteria	189918
Mycobacterium abscessus	Bacteria	561007
Mycobacterium avium	Bacteria	243243
Mycobacterium bovis	Bacteria	233413
Mycobacterium gilvum	Bacteria	350054
Mycobacterium leprae	Bacteria	561304
Mycobacterium marinum	Bacteria	216594
Mycobacterium smegmatis	Bacteria	246196
Mycobacterium tuberculosis	Bacteria	83331
Mycobacterium ulcerans	Bacteria	362242
Mycobacterium vanbaalenii	Bacteria	350058
Myxococcaceae Anaeromyxobacter	Bacteria	404589
Myxococcaceae Anaeromyxobacter	Bacteria	447217
Myxococcus xanthus	Bacteria	246197
Nakamurella multipartita	Bacteria	479431

Organism	Domain	TaxID
Nanoarchaeum equitans	Archaea	228908
Natranaerobius thermophilus	Bacteria	457570
Natrialba magadii	Archaea	547559
Natronomonas pharaonis	Archaea	348780
Nautilia profundicola	Bacteria	598659
Neisseria gonorrhoeae	Bacteria	242231
Neorickettsia risticii	Bacteria	434131
Neorickettsia sennetsu	Bacteria	222891
Nitrobacter hamburgensis	Bacteria	323097
Nitrobacter winogradskyi	Bacteria	323098
Nitrosococcus halophilus	Bacteria	472759
Nitrosococcus oceani	Bacteria	323261
Nitrosococcus watsonii	Bacteria	105559
Nitrosomonas europaea	Bacteria	228410
Nitrosomonas eutropha	Bacteria	335283
Nitrosopumilus maritimus	Archaea	436308
Nitrosospira multiformis	Bacteria	323848
Nitrospiraceae Nitrospira	Bacteria	330214
Nocardia farcinica	Bacteria	247156
Nocardioidaceae Nocardioides	Bacteria	196162
Nostocaceae Nostoc	Bacteria	103690
Nostoc punctiforme	Bacteria	63737
Novosphingobium aromaticivorans	Bacteria	279238
Oceanobacillus iheyensis	Bacteria	221109
Oceanospirillaceae Marinomonas	Bacteria	400668
Ochrobactrum anthropi	Bacteria	439375
Oenococcus oeni	Bacteria	203123
Oligotropha carboxidovorans	Bacteria	504832
Olsenella uli	Bacteria	633147
Opitutus terrae	Bacteria	452637
Orientia tsutsugamushi	Bacteria	357244
Oxalobacteraceae Herminiimonas	Bacteria	204773
Oxalobacteraceae Janthinobacterium	Bacteria	375286
Paenibacillaceae Paenibacillus	Bacteria	324057
Paenibacillaceae Paenibacillus	Bacteria	481743
Paenibacillus polymyxa	Bacteria	349520
Paludibacter propionicigenes	Bacteria	694427
Pantoea ananatis	Bacteria	706191
Pantoea vagans	Bacteria	712898
Parabacteroides distasonis	Bacteria	435591
Paracoccus denitrificans	Bacteria	318586

Organism	Domain	TaxID
Parvibaculum lavamentivorans	Bacteria	402881
Parvularcula bermudensis	Bacteria	314260
Pectobacterium atrosepticum	Bacteria	218491
Pectobacterium wasabiae	Bacteria	561231
Pediococcus pentosaceus	Bacteria	278197
Pedobacter heparinus	Bacteria	485917
Pelagibacter ubique	Bacteria	335992
Pelobacter carbinolicus	Bacteria	338963
Pelobacter propionicus	Bacteria	338966
Pelodictyon luteolum	Bacteria	319225
Pelodictyon phaeoclathratiforme	Bacteria	324925
Pelotomaculum thermopropionicum	Bacteria	370438
Persephonella marina	Bacteria	123214
Petrotoga mobilis	Bacteria	403833
Phenylobacterium zucineum	Bacteria	450851
Photobacterium profundum	Bacteria	298386
Phyllobacteriaceae Chelativorans	Bacteria	266779
Picrophilus torridus	Archaea	263820
Pirellula staleyi	Bacteria	530564
Planctomyces limnophilus	Bacteria	521674
Polaromonas naphthalenivorans	Bacteria	365044
Porphyromonas gingivalis	Bacteria	431947
Prevotella melaninogenica	Bacteria	553174
Prevotella ruminicola	Bacteria	264731
Prochlorococcus marinus	Bacteria	146891
Propionibacterium acnes	Bacteria	267747
Prosthecochloris aestuarii	Bacteria	290512
Proteobacteria Magnetococcus	Bacteria	156889
Proteus mirabilis	Bacteria	529507
Protochlamydia amoebophila	Bacteria	264201
Pseudoalteromonas atlantica	Bacteria	342610
Pseudoalteromonas haloplanktis	Bacteria	326442
Pseudomonas aeruginosa	Bacteria	557722
Pseudomonas entomophila	Bacteria	384676
Pseudomonas fluorescens	Bacteria	205922
Pseudomonas mendocina	Bacteria	399739
Pseudomonas putida	Bacteria	351746
Pseudomonas stutzeri	Bacteria	379731
Psychrobacter arcticus	Bacteria	259536
Psychrobacter cryohalolentis	Bacteria	335284
Psychromonas ingrahamii	Bacteria	357804

Organism	Domain	TaxID
Puniceispirillum marinum	Bacteria	488538
pv. campestris	Bacteria	314565
pv. citri	Bacteria	190486
pv. oryzae	Bacteria	291331
pv. syringae	Bacteria	205918
Pyrobaculum aerophilum	Archaea	178306
Pyrobaculum arsenaticum	Archaea	340102
Pyrobaculum calidifontis	Archaea	410359
Pyrobaculum islandicum	Archaea	384616
Pyrococcus abyssi	Archaea	272844
Pyrococcus furiosus	Archaea	186497
Pyrococcus horikoshii	Archaea	70601
Ralstonia pickettii	Bacteria	428406
Ralstonia solanacearum	Bacteria	859656
Renibacterium salmoninarum	Bacteria	288705
Rhizobium etli	Bacteria	347834
Rhodobacteraceae Jannaschia	Bacteria	290400
Rhodobacteraceae Ruegeria	Bacteria	292414
Rhodobacter capsulatus	Bacteria	272942
Rhodobacter sphaeroides	Bacteria	272943
Rhodococcus equi	Bacteria	685727
Rhodococcus erythropolis	Bacteria	234621
Rhodococcus jostii	Bacteria	101510
Rhodococcus opacus	Bacteria	632772
Rhodocyclaceae Azoarcus	Bacteria	62928
Rhodocyclaceae Thauera	Bacteria	85643
Rhodomicrobium vannielii	Bacteria	648757
Rhodopirellula baltica	Bacteria	243090
Rhodopseudomonas palustris	Bacteria	316055
Rhodospirillaceae Azospirillum	Bacteria	137722
Rhodospirillum centenum	Bacteria	414684
Rhodospirillum rubrum	Bacteria	269796
Rhodothermus marinus	Bacteria	518766
Rickettsia africae	Bacteria	347255
Rickettsia akari	Bacteria	293614
Rickettsia bellii	Bacteria	391896
Rickettsia canadensis	Bacteria	293613
Rickettsia conorii	Bacteria	272944
Rickettsia felis	Bacteria	315456
Rickettsia massiliae	Bacteria	416276
Rickettsia peacockii	Bacteria	562019

Organism	Domain	TaxID
Rickettsia prowazekii	Bacteria	272947
Rickettsia rickettsii	Bacteria	392021
Rickettsia typhi	Bacteria	257363
Riemerella anatipestifer	Bacteria	693978
Riesia pediculicola	Bacteria	515618
Robiginitalea biformata	Bacteria	313596
Roseiflexus castenholzii	Bacteria	383372
Roseobacter denitrificans	Bacteria	375451
Rothia dentocariosa	Bacteria	762948
Rothia mucilaginosa	Bacteria	680646
Rubrobacter xylanophilus	Bacteria	266117
Ruegeria pomeroyi	Bacteria	246200
Ruthia magnifica	Bacteria	413404
Saccharomonospora viridis	Bacteria	471857
Saccharophagus degradans	Bacteria	203122
Saccharopolyspora erythraea	Bacteria	405948
Salinibacter ruber	Bacteria	309807
Salinispora arenicola	Bacteria	391037
Salinispora tropica	Bacteria	369723
Sanguibacter keddieii	Bacteria	446469
Sebaldella termitidis	Bacteria	526218
Sedis Exiguobacterium	Bacteria	360911
Segniliparus rotundus	Bacteria	640132
serogroup 1	Bacteria	423212
serogroup C	Bacteria	374833
serogroup V	Bacteria	208435
serotype M1	Bacteria	160490
serovar 3	Bacteria	434271
serovar 62:z4,z23:–	Bacteria	882884
serovar Copenhageni	Bacteria	267671
serovar Hardjo-bovis	Bacteria	355277
serovar Patoc	Bacteria	355278
Serratia proteamaculans	Bacteria	399741
Shewanella amazonensis	Bacteria	326297
Shewanella baltica	Bacteria	325240
Shewanellaceae Shewanella	Bacteria	351745
Shewanellaceae Shewanella	Bacteria	60480
Shewanellaceae Shewanella	Bacteria	94122
Shewanella denitrificans	Bacteria	318161
Shewanella frigidimarina	Bacteria	318167
Shewanella halifaxensis	Bacteria	458817
Shewanella loihica	Bacteria	323850

Organism	Domain	TaxID
Shewanella oneidensis	Bacteria	211586
Shewanella pealeana	Bacteria	398579
Shewanella piezotolerans	Bacteria	225849
Shewanella putrefaciens	Bacteria	319224
Shewanella sediminis	Bacteria	425104
Shewanella violacea	Bacteria	637905
Shewanella woodyi	Bacteria	392500
Shigella boydii	Bacteria	344609
Shigella dysenteriae	Bacteria	300267
Shigella sonnei	Bacteria	300269
Sideroxydans lithotrophicus	Bacteria	580332
Sinorhizobium fredii	Bacteria	394
Sinorhizobium medicae	Bacteria	366394
Sinorhizobium meliloti	Bacteria	266834
Slackia heliotrinireducens	Bacteria	471855
Sodalis glossinidius	Bacteria	343509
Solibacter usitatus	Bacteria	234267
Sorangium cellulosum	Bacteria	448385
Sphaerobacter thermophilus	Bacteria	479434
Sphingobium japonicum	Bacteria	452662
Sphingomonas wittichii	Bacteria	392499
Sphingopyxis alaskensis	Bacteria	317655
Spirochaeta smaragdinae	Bacteria	573413
Spirochaeta thermophila	Bacteria	665571
Spirosoma linguale	Bacteria	504472
Stackebrandtia nassauensis	Bacteria	446470
Staphylococcus epidermidis	Bacteria	176280
Staphylococcus haemolyticus	Bacteria	279808
Staphylococcus lugdunensis	Bacteria	698737
Staphylothermus hellenicus	Archaea	591019
Staphylothermus marinus	Archaea	399550
Starkeya novella	Bacteria	639283
Stenotrophomonas maltophilia	Bacteria	522373
Stolbur group	Bacteria	59748
str. Challis	Bacteria	467705
Streptobacillus moniliformis	Bacteria	519441
Streptococcus gallolyticus	Bacteria	637909
Streptococcus mitis	Bacteria	365659
Streptococcus mutans	Bacteria	511691
Streptococcus pneumoniae	Bacteria	189423
Streptococcus sanguinis	Bacteria	388919

Organism	Domain	TaxID
Streptococcus suis	Bacteria	391295
Streptococcus thermophilus	Bacteria	299768
Streptococcus uberis	Bacteria	218495
Streptomyces avermitilis	Bacteria	227882
Streptomyces coelicolor	Bacteria	100226
Streptomyces scabiei	Bacteria	680198
Streptosporangium roseum	Bacteria	479432
subsp. acidocaldarius	Bacteria	521098
subsp. asymbioticus	Bacteria	312153
subsp. aureus	Bacteria	93062
subsp. bulgaricus	Bacteria	390333
subsp. carnosus	Bacteria	396513
subsp. carotovorum	Bacteria	561230
subsp. cloacae	Bacteria	716541
subsp. cremoris	Bacteria	416870
subsp. dassonvillei	Bacteria	446468
subsp. desulfuricans	Bacteria	525146
subsp. enterocolitica	Bacteria	393305
subsp. equi	Bacteria	553482
subsp. equisimilis	Bacteria	486410
subsp. fetus	Bacteria	360106
subsp. griseus	Bacteria	455632
subsp. hydrophila	Bacteria	380703
subsp. indica	Bacteria	395963
subsp. jejuni	Bacteria	407148
subsp. lactis	Bacteria	442563
subsp. laumondii	Bacteria	243265
subsp. longum	Bacteria	890402
subsp. mathranii	Bacteria	583358
subsp. mesenteroides	Bacteria	203120
subsp. michiganensis	Bacteria	443906
subsp. mobilis	Bacteria	622759
subsp. multocida	Bacteria	272843
subsp. nucleatum	Bacteria	190304
subsp. pallidum	Bacteria	243276
subsp. sakei	Bacteria	314315
subsp. salmonicida	Bacteria	382245
subsp. saprophyticus	Bacteria	342451
subsp. shermanii	Bacteria	754252
subsp. subtilis	Bacteria	224308
subsp. succinogenes	Bacteria	59374

Organism	Domain	TaxID
subsp. tengcongensis	Bacteria	273068
subsp. tularensis	Bacteria	393115
subsp. wolfei	Bacteria	335541
subsp. xyli	Bacteria	281090
Sulcia muelleri	Bacteria	706194
Sulfolobus acidocaldarius	Archaea	330779
Sulfolobus islandicus	Archaea	425944
Sulfolobus solfataricus	Archaea	273057
Sulfolobus tokodaii	Archaea	273063
Sulfurihydrogenibium azorense	Bacteria	204536
Sulfurimonas autotrophica	Bacteria	563040
Sulfurimonas denitrificans	Bacteria	326298
Sulfurospirillum deleyianum	Bacteria	525898
Symbiobacterium thermophilum	Bacteria	292459
Synechococcus elongatus	Bacteria	269084
Syntrophobacter fumaroxidans	Bacteria	335543
Syntrophothermus lipocalidus	Bacteria	643648
Syntrophus aciditrophicus	Bacteria	56780
Teredinibacter turnerae	Bacteria	377629
testosteroni CNB-1	Bacteria	688245
Thermanaerovibrio acidaminovorans	Bacteria	525903
Thermincola potens	Bacteria	635013
Thermoanaerobacteraceae Thermoanaerobacter	Bacteria	399726
Thermoanaerobacteraceae Thermoanaerobacter	Bacteria	573062
Thermoanaerobacter italicus	Bacteria	580331
Thermoanaerobacterium thermosaccharolyticum	Bacteria	580327
Thermoanaerobacter pseudethanolicus	Bacteria	340099
Thermobaculum terrenum	Bacteria	525904
Thermobifida fusca	Bacteria	269800
Thermobispora bispora	Bacteria	469371
Thermococcus gammatolerans	Archaea	593117
Thermococcus kodakarensis	Archaea	69014
Thermococcus onnurineus	Archaea	523850
Thermococcus sibiricus	Archaea	604354
Thermocrinis albus	Bacteria	638303
Thermodesulfovibrio yellowstonii	Bacteria	289376
Thermofilum pendens	Archaea	368408
Thermomicrobium roseum	Bacteria	309801
Thermomonospora curvata	Bacteria	471852
Thermoplasma acidophilum	Archaea	273075
Thermoplasma volcanium	Archaea	273116

Organism	Domain	TaxID
Thermoproteus neutrophilus	Archaea	444157
Thermosediminibacter oceani	Bacteria	555079
Thermosipho africanus	Bacteria	484019
Thermosipho melanesiensis	Bacteria	391009
Thermosphaera aggregans	Archaea	633148
Thermosynechococcus elongatus	Bacteria	197221
Thermotogaceae Thermotoga	Bacteria	126740
Thermotoga lettingae	Bacteria	416591
Thermotoga maritima	Bacteria	243274
Thermotoga naphthophila	Bacteria	590168
Thermotoga neapolitana	Bacteria	309803
Thermotoga petrophila	Bacteria	390874
Thermus thermophilus	Bacteria	262724
Thiobacillus denitrificans	Bacteria	292415
Thiomicrospira crunogena	Bacteria	317025
Thiomonas intermedia	Bacteria	75379
Tolumonas auensis	Bacteria	595494
Treponema denticola	Bacteria	243275
Trichodesmium erythraeum	Bacteria	203124
Tropheryma whipplei	Bacteria	218496
Truepera radiovictrix	Bacteria	649638
Tsukamurella paurometabola	Bacteria	521096
unclassified Chroococcales	Bacteria	713887
unclassified Flavobacteriaceae	Bacteria	531844
Variovorax paradoxus	Bacteria	543728
Veillonella parvula	Bacteria	479436
Verminephrobacter eiseniae	Bacteria	391735
Vibrio cholerae	Bacteria	579112
Vibrio harveyi	Bacteria	338187
Vibrionaceae Vibrio	Bacteria	150340
Vibrio parahaemolyticus	Bacteria	223926
Vibrio splendidus	Bacteria	575788
Vibrio vulnificus	Bacteria	216895
Vulcanisaeta distributa	Archaea	572478
Waddlia chondrophila	Bacteria	716544
Wigglesworthia glossinidia	Bacteria	36870
witches'-broom phytoplasma	Bacteria	322098
Wolbachieae Wolbachia	Bacteria	66084
Wolinella succinogenes	Bacteria	273121
Xanthobacter autotrophicus	Bacteria	78245
Xanthomonadaceae Xanthomonas	Bacteria	29447

Organism	**Domain**	**TaxID**
Xenorhabdus bovienii	Bacteria	406818
Xenorhabdus nematophila	Bacteria	406817
Xylanimonas cellulosilytica	Bacteria	446471
Xylella fastidiosa	Bacteria	160492
yellows phytoplasma	Bacteria	262768
Yersinia pestis	Bacteria	349746
Yersinia pseudotuberculosis	Bacteria	349747
Zunongwangia profunda	Bacteria	655815

Appendix C

Source code

ncbi_process.pl :

```
#!/usr/bin/perl
#
# Take a bunch of directories containing NCBI data & process into
    counts.

$code_base = "/Users/dtwright/Research/Codon_Bias/code";

$dir_list = $ARGV[0];
if(! -f $dir_list) {
  die "Please provide a list of directories to process.\n";
}

open(DIRS,"<",$dir_list);

DIR: while(<DIRS>) {
        chomp;
        $dir = $_;
  @parts = split(/_/,$dir);
  $current_sp = $parts[0]."_".$parts[1];
  chdir($dir);
  $taxid = "";
  $spname = "";
  $len = 0;
  $need_other = 0;
  $file_base = "";
  foreach $rpt (<*.rpt>) {
    # find the seq file with the most data in it; this will be
    # the main genome data.
    open(RPT, "<", $rpt);
    while(<RPT>) {
      if(/length.=.(\d+)/) {
        if($1 > $len) {
          $len = $1;
```

```
            $need_other = 1;
            $file_base = (split(/\./,$rpt))[0];
          }
          else {
            $need_other = 0;
          }
        }
        if($need_other = 1) {
          if(/^Taxid:.(\d+)/) {
            $taxid = $1;
          }
          if(/^Taxname:.(.+)/) {
            $spname = $1;
          }
          if(/^Genetic Code:.(\d+)/) {
            $gcode = $1;
            if($gcode != 11 && $gcode != 1) {
              print "Skipping $dir - non-standard code\n";
              chdir("..");
              next DIR;
            }
          }
        }
      }
    }
    print("Found: $dir, $file_base (len - $len, taxid - $taxid)\n");
    print("Processing...");
    $call = "perl ".$code_base."/make_counts_from_ncbi.pl ".$file_base."
       ffn $taxid \"$spname\" 1>>../ncbi_counts.list";
    # print($call);
    if(system($call) != 0) {
      die("Error with exec $call\n");
    }
    print("done.\n");
    chdir("..");

    print("current: $current_sp, prev: $prev_sp\n");
    $prev_sp = $current_sp;
}
```

make_counts_from_ncbi.pl :

```
#!/usr/bin/perl
#
```

```
# Takes a .spsum file from http://www.kazusa.or.jp/codon/ and turns it into %
# counts for each codon.
#
# TODO: figure out GC/AT proportions from this data -- should be easy.

$fname = $ARGV[0];
if(! -f $fname) {
  die("Please provide an input file name.\nUsage: make_counts_from_ncbi <input filename> <taxid> <species name>\n");
}

$taxid = $ARGV[1];
$spname = $ARGV[2];

if($taxid eq "" || $spname eq "") {
  die("Usage: make_counts_from_ncbi <input filename> <taxid> <species name>\n");
}

@codon_order = ("CGA","CGC","CGG","CGU","AGA","AGG","CUA","CUC","CUG","CUU","UUA","UUG","UCA","UCC","UCG","UCU","AGC","AGU","ACA","ACC","ACG","ACU","CCA","CCC","CCG","CCU","GCA","GCC","GCG","GCU","GGA","GGC","GGG","GGU","GUA","GUC","GUG","GUU","AAA","AAG","AAC","AAU","CAA","CAG","CAC","CAU","GAA","GAG","GAC","GAU","UAC","UAU","UGC","UGU","UUC","UUU","AUA","AUC","AUU","AUG","UGG","UAA","UAG","UGA");

%codemap = ( "UUU" => "Phe", "UUC" => "Phe", "UUA" => "Leu", "UUG" => "Leu", "CUU" => "Leu",
             "CUC" => "Leu", "CUA" => "Leu", "CUG" => "Leu", "AUU" => "Ile", "AUC" => "Ilu",
             "AUA" => "Ile", "AUG" => "Met", "GUU" => "Val", "GUC" => "Val", "GUA" => "Val",
             "GUG" => "Val", "UCU" => "Ser", "UCC" => "Ser", "UCA" => "Ser", "UCG" => "Ser",
             "CCU" => "Pro", "CCC" => "Pro", "CCA" => "Pro", "CCG" => "Pro", "ACU" => "Thr",
             "ACC" => "Thr", "ACA" => "Thr", "ACG" => "Thr", "GCU" => "Ala", "GCC" => "Ala",
             "GCA" => "Ala", "GCG" => "Ala", "UAU" => "Tyr", "UAC" => "Tyr", "UAA" => "STP",
             "UAG" => "STP", "CAU" => "His", "CAC" => "His", "CAA" => "Gln", "CAG" => "Gln",
             "AAU" => "Asn", "AAC" => "Asn", "AAA" => "Lys", "AAG" => "Lys", "GAU" => "Asp",
```

```
                "GAC" => "Asp", "GAA" => "Glu", "GAG" => "Glu", "UGU" => "
                   Cys", "UGC" => "Cys",
32              "UGA" => "STP", "UGG" => "Trp", "CGU" => "Arg", "CGC" => "
                   Arg", "CGA" => "Arg",
                "CGG" => "Arg", "AGU" => "Ser", "AGC" => "Ser", "AGA" => "
                   Arg", "AGG" => "Arg",
34              "GGU" => "Gly", "GGC" => "Gly", "GGA" => "Gly", "GGG" => "
                   Gly" );

36 %revcodemap = ( );
   foreach $key (keys %codemap) {
38   push @{ $revcodemap{$codemap{$key}} }, $key;
   }
40
   # diagnostic printouts for arrays
42 # foreach $key (keys %codemap) {
     # print "$key => $codemap{$key}\n";
44 # }

46 # foreach $key (keys %revcodemap) {
     # print "$key =>";
48   # foreach $val (@{ $revcodemap{$key} }) {
       # print " $val";
50   # }
     # print "\n";
52 # }

54 open(IN, "<", $fname) or die $!;

56 %count_data = ( );
   foreach $key (keys %codemap) {
58   $count_data{$key} = 0;
   }
60
   $cur_gene = "";
62 $gene_count = 0;
   while(<IN>) {
64   chomp;
     if(/^>ref/) {
66     # save data
       if($cur_gene ne "") {
68       # fix T/U problem
         $cur_gene =~ s/T/U/g;
70       $len = length($cur_gene);
         # figure out how many codons
```

```
72        $codons = int($len/3);
          $leftover = $len%3;
74                        if($leftover != 0) {
                                print STDERR "uneven codon count at $_\
                                    n";
76                        }
          $fmt_count = $codons+$leftover;
78        # make format string for unpack()
          $unpack_fmt = "";
80        for($i=0;$i<$fmt_count;$i++) {
            $unpack_fmt .= "a3";
82        }
          # extract codons
84        @codons = unpack($unpack_fmt,$cur_gene);
          # save count
86        foreach $codon (@codons) {
            $count_data{$codon}++;
88        }
          # clear accumulator
90        $cur_gene = "";

92                        # count gene
                          $gene_count++;
94      }
    }
96  else {
      # This line has actual coding data; save it. we have to merge lines
          b/c of 71 char line width.
98    $cur_gene .= $_;
    }
100 }

102 # process remaining data from last gene
    if($cur_gene ne "") {
104   # fix T/U problem
      $cur_gene =~ s/T/U/g;
106   $len = length($cur_gene);
      # figure out how many codons
108   $codons = int($len/3);
      $leftover = $len%3;
110   $fmt_count = $codons+$leftover;
      # make format string for unpack()
112   $unpack_fmt = "";
      for($i=0;$i<$fmt_count;$i++) {
114     $unpack_fmt .= "a3";
```

```
    }
    # extract codons
    @codons = unpack($unpack_fmt,$cur_gene);
    # save count
    foreach $codon (@codons) {
      $count_data{$codon}++;
    }
    # clear accumulator
    $cur_gene = "";
}

# $codon_sum = 0;
# foreach $codon (@codon_order) {
#         # $codon_sum += $count_data{$codon};
# }

# print header line
print("$taxid:$spname: $gene_count\n");

foreach $codon (@codon_order) {
  print($count_data{$codon}." ");
}
print("\n");
```

make_bias_counts.pl :

```
#!/usr/bin/perl
#
# Takes a .spsum file from http://www.kazusa.or.jp/codon/ and turns it
    into %
# counts for each codon.
#
# TODO: figure out GC/AT proportions from this data -- should be easy.

$fname = $ARGV[0];
if(! -f $fname) {
  die("Please provide an input file name.\n");
}

@codons = ("CGA","CGC","CGG","CGU","AGA","AGG","CUA","CUC","CUG","CUU",
    "UUA","UUG","UCA","UCC","UCG","UCU","AGC","AGU","ACA","ACC","ACG","
    ACU","CCA","CCC","CCG","CCU","GCA","GCC","GCG","GCU","GGA","GGC","
    GGG","GGU","GUA","GUC","GUG","GUU","AAA","AAG","AAC","AAU","CAA","
    CAG","CAC","CAU","GAA","GAG","GAC","GAU","UAC","UAU","UGC","UGU","
```

```
    UUC","UUU","AUA","AUC","AUU","AUG","UGG","UAA","UAG","UGA");

%codemap = ( "UUU" => "Phe", "UUC" => "Phe", "UUA" => "Leu", "UUG" => "
    Leu", "CUU" => "Leu",
                "CUC" => "Leu", "CUA" => "Leu", "CUG" => "Leu", "AUU" => "
                   Ile", "AUC" => "Ilu",
                "AUA" => "Ile", "AUG" => "Met", "GUU" => "Val", "GUC" => "
                   Val", "GUA" => "Val",
                "GUG" => "Val", "UCU" => "Ser", "UCC" => "Ser", "UCA" => "
                   Ser", "UCG" => "Ser",
                "CCU" => "Pro", "CCC" => "Pro", "CCA" => "Pro", "CCG" => "
                   Pro", "ACU" => "Thr",
                "ACC" => "Thr", "ACA" => "Thr", "ACG" => "Thr", "GCU" => "
                   Ala", "GCC" => "Ala",
                "GCA" => "Ala", "GCG" => "Ala", "UAU" => "Tyr", "UAC" => "
                   Tyr", "UAA" => "STP",
                "UAG" => "STP", "CAU" => "His", "CAC" => "His", "CAA" => "
                   Gln", "CAG" => "Gln",
                "AAU" => "Asn", "AAC" => "Asn", "AAA" => "Lys", "AAG" => "
                   Lys", "GAU" => "Asp",
                "GAC" => "Asp", "GAA" => "Glu", "GAG" => "Glu", "UGU" => "
                   Cys", "UGC" => "Cys",
                "UGA" => "STP", "UGG" => "Trp", "CGU" => "Arg", "CGC" => "
                   Arg", "CGA" => "Arg",
                "CGG" => "Arg", "AGU" => "Ser", "AGC" => "Ser", "AGA" => "
                   Arg", "AGG" => "Arg",
                "GGU" => "Gly", "GGC" => "Gly", "GGA" => "Gly", "GGG" => "
                   Gly" );

%revcodemap = ( );
foreach $key (keys %codemap) {
  push @{ $revcodemap{$codemap{$key}} }, $key;
}

# diagnostic printouts for arrays
# foreach $key (keys %codemap) {
  # print "$key => $codemap{$key}\n";
# }

# foreach $key (keys %revcodemap) {
  # print "$key =>";
  # foreach $val (@{ $revcodemap{$key} }) {
    # print " $val";
  # }
  # print "\n";
```

```
# }

open(IN, "<", $fname) or die $!;

%orgs_data = ( );

while(<IN>) {
  chomp;
  if(/(.+):(.+): (\d+)/) {
    $taxid = $1;
    $spname = $2;
    $cdscount = $3;
    # print "----\n$taxid $spname $cdscount\n";
    $orgs_data{$taxid}->{"spname"} = $spname;
    $orgs_data{$taxid}->{"cdscount"} = $cdscount;

    #print $orgs_data{$taxid}->{"spname"}."\n";
    #print $orgs_data{$taxid}->{"cdscount"}."\n";
  }
  else {
    # Split line into codons
    @codon_counts = split( / /);
    $total_codons = 0;
    foreach $count (@codon_counts) {
      $total_codons += $count;
    }
    if($#codons != $#codon_counts) {
      print "Bad codon count - doesn't match labels at $taxid\n";
      exit;
    }
    for($i=0;$i<=$#codons;$i++) {
      my $cur_codon = $codons[$i];
      $orgs_data{$taxid}->{"data"}->{$cur_codon} = $codon_counts[$i]/
          $total_codons;
    }
  }
}

# print out the data arrays - SCILAB/MATLAB format
#foreach $key (keys %orgs_data) {
  #print "DataMat".$key." = ".
#"[ ".$orgs_data{$key}->{'data'}->{'AAA'}." ".$orgs_data{$key}->{'data
    '}->{'UAA'}." ".$orgs_data{$key}->{'data'}->{'GAA'}." ".$orgs_data{
    $key}->{'data'}->{'CAA'}."; ".
```

```
#$orgs_data{$key}->{'data'}->{'AUA'}." ".$orgs_data{$key}->{'data'}->{'
    UUA'}." ".$orgs_data{$key}->{'data'}->{'GUA'}." ".$orgs_data{$key
    }->{'data'}->{'CUA'}."; ".
87 #$orgs_data{$key}->{'data'}->{'AGA'}." ".$orgs_data{$key}->{'data'}->{'
    UGA'}." ".$orgs_data{$key}->{'data'}->{'GGA'}." ".$orgs_data{$key
    }->{'data'}->{'CGA'}."; ".
   #$orgs_data{$key}->{'data'}->{'ACA'}." ".$orgs_data{$key}->{'data'}->{'
    UCA'}." ".$orgs_data{$key}->{'data'}->{'GCA'}." ".$orgs_data{$key
    }->{'data'}->{'CCA'}." /\n";
89 #
   #print "DataMat".$key."(:,:,2) = ".
91 #"[ ".$orgs_data{$key}->{'data'}->{'AAU'}." ".$orgs_data{$key}->{'data
    '}->{'UAU'}." ".$orgs_data{$key}->{'data'}->{'GAU'}." ".$orgs_data{
    $key}->{'data'}->{'CAU'}."; ".
   #$orgs_data{$key}->{'data'}->{'AUU'}." ".$orgs_data{$key}->{'data'}->{'
    UUU'}." ".$orgs_data{$key}->{'data'}->{'GUU'}." ".$orgs_data{$key
    }->{'data'}->{'CUU'}."; ".
93 #$orgs_data{$key}->{'data'}->{'AGU'}." ".$orgs_data{$key}->{'data'}->{'
    UGU'}." ".$orgs_data{$key}->{'data'}->{'GGU'}." ".$orgs_data{$key
    }->{'data'}->{'CGU'}."; ".
   #$orgs_data{$key}->{'data'}->{'ACU'}." ".$orgs_data{$key}->{'data'}->{'
    UCU'}." ".$orgs_data{$key}->{'data'}->{'GCU'}." ".$orgs_data{$key
    }->{'data'}->{'CCU'}." /\n";
95 #
   #print "DataMat".$key."(:,:,3) = ".
97 #"[ ".$orgs_data{$key}->{'data'}->{'AAG'}." ".$orgs_data{$key}->{'data
    '}->{'UAG'}." ".$orgs_data{$key}->{'data'}->{'GAG'}." ".$orgs_data{
    $key}->{'data'}->{'CAG'}."; ".
   #$orgs_data{$key}->{'data'}->{'AUG'}." ".$orgs_data{$key}->{'data'}->{'
    UUG'}." ".$orgs_data{$key}->{'data'}->{'GUG'}." ".$orgs_data{$key
    }->{'data'}->{'CUG'}."; ".
99 #$orgs_data{$key}->{'data'}->{'AGG'}." ".$orgs_data{$key}->{'data'}->{'
    UGG'}." ".$orgs_data{$key}->{'data'}->{'GGG'}." ".$orgs_data{$key
    }->{'data'}->{'CGG'}."; ".
   #$orgs_data{$key}->{'data'}->{'ACG'}." ".$orgs_data{$key}->{'data'}->{'
    UCG'}." ".$orgs_data{$key}->{'data'}->{'GCG'}." ".$orgs_data{$key
    }->{'data'}->{'CCG'}." /\n";
101 #
   #print "DataMat".$key."(:,:,4) = ".
103 #"[ ".$orgs_data{$key}->{'data'}->{'AAC'}." ".$orgs_data{$key}->{'data
    '}->{'UAC'}." ".$orgs_data{$key}->{'data'}->{'GAC'}." ".$orgs_data{
    $key}->{'data'}->{'CAC'}."; ".
   #$orgs_data{$key}->{'data'}->{'AUC'}." ".$orgs_data{$key}->{'data'}->{'
    UUC'}." ".$orgs_data{$key}->{'data'}->{'GUC'}." ".$orgs_data{$key
    }->{'data'}->{'CUC'}."; ".
```

```
105 #$orgs_data{$key}->{'data'}->{'AGC'}." ".$orgs_data{$key}->{'data'}->{'
    UGC'}." ".$orgs_data{$key}->{'data'}->{'GGC'}." ".$orgs_data{$key
    }->{'data'}->{'CGC'}."; ".
    #$orgs_data{$key}->{'data'}->{'ACC'}." ".$orgs_data{$key}->{'data'}->{'
    UCC'}." ".$orgs_data{$key}->{'data'}->{'GCC'}." ".$orgs_data{$key
    }->{'data'}->{'CCC'}." /\n";
107 #
     #print "\n\n";
109 #}

111 # print out the data arrays - R format
    $num = scalar(keys(%orgs_data));
113 $rownames = "c(";
    $start = 1;
115 foreach $key (keys %orgs_data) {
      $name = $key;
117   $name =~ /^(\d+)/;
            $name = $1;
119   #$name =~ s/:/_/g;
      #$name =~ s/'//g;
121   #$name =~ s/,//g;
      #$name =~ s/\.//g;
123   if($start != 1) {
        $rownames .= ",";
125   } else {
        $start = 0;
127   }
      $rownames .= "\"$name\"";
129 }
    $rownames .= ")";
131
    $colnames = "c(";
133 $start = 1;
    foreach $codon (@codons) {
135   if($start != 1) {
        $colnames .= ",";
137   } else {
        $start = 0;
139   }
      $colnames .= "\"$codon\"";
141 }
    $colnames .= ")";
143
    # Set up R matrix
```

```
print("BiasData <- matrix(nrow=$num,ncol=64,dimnames=list($rownames,
    $colnames))\n");

# ...and print data.
foreach $key (keys %orgs_data) {
  $name = $key;
  $name =~ /^(\d+)/;
        $name = $1;
        foreach $codon (@codons) {
                print("BiasData[\"$name\",\"$codon\"] <- ".$orgs_data{
                    $key}->{'data'}->{$codon}."\n");
        }
}

#foreach $key (keys %orgs_data) {
  #$name = $key;
  #$name =~ s/ /_/g;
  #$name =~ s/:/_/g;
  #$name =~ s/'//g;
  #$name =~ s/,//g;
  #$name =~ s/\.//g;
  #print "BiasData.Msp\$tax".$name." <- array(c(".
#$orgs_data{$key}->{'data'}->{'AAA'}.",".$orgs_data{$key}->{'data'}->{'
    UAA'}.",".$orgs_data{$key}->{'data'}->{'GAA'}.",\n".
#$orgs_data{$key}->{'data'}->{'CAA'}.",".$orgs_data{$key}->{'data'}->{'
    AUA'}.",".$orgs_data{$key}->{'data'}->{'UUA'}.",\n".
#$orgs_data{$key}->{'data'}->{'GUA'}.",".$orgs_data{$key}->{'data'}->{'
    CUA'}.",".$orgs_data{$key}->{'data'}->{'AGA'}.",\n".
#$orgs_data{$key}->{'data'}->{'UGA'}.",".$orgs_data{$key}->{'data'}->{'
    GGA'}.",".$orgs_data{$key}->{'data'}->{'CGA'}.",\n".
#$orgs_data{$key}->{'data'}->{'ACA'}.",".$orgs_data{$key}->{'data'}->{'
    UCA'}.",".$orgs_data{$key}->{'data'}->{'GCA'}.",\n".
#$orgs_data{$key}->{'data'}->{'CCA'}.",".$orgs_data{$key}->{'data'}->{'
    AAU'}.",".$orgs_data{$key}->{'data'}->{'UAU'}.",\n".
#$orgs_data{$key}->{'data'}->{'GAU'}.",".$orgs_data{$key}->{'data'}->{'
    CAU'}.",".$orgs_data{$key}->{'data'}->{'AUU'}.",\n".
#$orgs_data{$key}->{'data'}->{'UUU'}.",".$orgs_data{$key}->{'data'}->{'
    GUU'}.",".$orgs_data{$key}->{'data'}->{'CUU'}.",\n".
#$orgs_data{$key}->{'data'}->{'AGU'}.",".$orgs_data{$key}->{'data'}->{'
    UGU'}.",".$orgs_data{$key}->{'data'}->{'GGU'}.",\n".
#$orgs_data{$key}->{'data'}->{'CGU'}.",".$orgs_data{$key}->{'data'}->{'
    ACU'}.",".$orgs_data{$key}->{'data'}->{'UCU'}.",\n".
#$orgs_data{$key}->{'data'}->{'GCU'}.",".$orgs_data{$key}->{'data'}->{'
    CCU'}.",".$orgs_data{$key}->{'data'}->{'AAG'}.",\n".
```

```
#$orgs_data{$key}->{'data'}->{'UAG'}.",".$orgs_data{$key}->{'data'}->{'
    GAG'}.",".$orgs_data{$key}->{'data'}->{'CAG'}.",\n".
#$orgs_data{$key}->{'data'}->{'AUG'}.",".$orgs_data{$key}->{'data'}->{'
    UUG'}.",".$orgs_data{$key}->{'data'}->{'GUG'}.",\n".
#$orgs_data{$key}->{'data'}->{'CUG'}.",".$orgs_data{$key}->{'data'}->{'
    AGG'}.",".$orgs_data{$key}->{'data'}->{'UGG'}.",\n".
#$orgs_data{$key}->{'data'}->{'GGG'}.",".$orgs_data{$key}->{'data'}->{'
    CGG'}.",".$orgs_data{$key}->{'data'}->{'ACG'}.",\n".
#$orgs_data{$key}->{'data'}->{'UCG'}.",".$orgs_data{$key}->{'data'}->{'
    GCG'}.",".$orgs_data{$key}->{'data'}->{'CCG'}.",\n".
#$orgs_data{$key}->{'data'}->{'AAC'}.",".$orgs_data{$key}->{'data'}->{'
    UAC'}.",".$orgs_data{$key}->{'data'}->{'GAC'}.",\n".
#$orgs_data{$key}->{'data'}->{'CAC'}.",".$orgs_data{$key}->{'data'}->{'
    AUC'}.",".$orgs_data{$key}->{'data'}->{'UUC'}.",\n".
#$orgs_data{$key}->{'data'}->{'GUC'}.",".$orgs_data{$key}->{'data'}->{'
    CUC'}.",".$orgs_data{$key}->{'data'}->{'AGC'}.",\n".
#$orgs_data{$key}->{'data'}->{'UGC'}.",".$orgs_data{$key}->{'data'}->{'
    GGC'}.",".$orgs_data{$key}->{'data'}->{'CGC'}.",\n".
#$orgs_data{$key}->{'data'}->{'ACC'}.",".$orgs_data{$key}->{'data'}->{'
    UCC'}.",".$orgs_data{$key}->{'data'}->{'GCC'}.",\n".
#$orgs_data{$key}->{'data'}->{'CCC'}."),c(4,4,4),".
#"dimnames=list(c(\"A\",\"U\",\"G\",\"C\"),c(\"A\",\"U\",\"G\",\"C\"),c
    (\"A\",\"U\",\"G\",\"C\")))\n";

  #print "\n\n";
#}
```

make_bias_counts_grouped.pl :

```
#!/usr/bin/perl
#
# Takes a .spsum file from http://www.kazusa.or.jp/codon/ and turns it
    into %
# counts for each codon.
#
# TODO: figure out GC/AT proportions from this data -- should be easy.

$fname = $ARGV[0];
if(! -f $fname) {
  die("Please␣provide␣an␣input␣file␣name.\n");
}

@codons = ("CGA","CGC","CGG","CGU","AGA","AGG","CUA","CUC","CUG","CUU",
    "UUA","UUG","UCA","UCC","UCG","UCU","AGC","AGU","ACA","ACC","ACG","
```

```
    ACU","CCA","CCC","CCG","CCU","GCA","GCC","GCG","GCU","GGA","GGC","
    GGG","GGU","GUA","GUC","GUG","GUU","AAA","AAG","AAC","AAU","CAA","
    CAG","CAC","CAU","GAA","GAG","GAC","GAU","UAC","UAU","UGC","UGU","
    UUC","UUU","AUA","AUC","AUU","AUG","UGG","UAA","UAG","UGA");

%codemap = ( "UUU" => "Phe", "UUC" => "Phe", "UUA" => "Leu", "UUG" => "
    Leu", "CUU" => "Leu",
             "CUC" => "Leu", "CUA" => "Leu", "CUG" => "Leu", "AUU" => "
                Ile", "AUC" => "Ilu",
             "AUA" => "Ile", "AUG" => "Met", "GUU" => "Val", "GUC" => "
                Val", "GUA" => "Val",
             "GUG" => "Val", "UCU" => "Ser", "UCC" => "Ser", "UCA" => "
                Ser", "UCG" => "Ser",
             "CCU" => "Pro", "CCC" => "Pro", "CCA" => "Pro", "CCG" => "
                Pro", "ACU" => "Thr",
             "ACC" => "Thr", "ACA" => "Thr", "ACG" => "Thr", "GCU" => "
                Ala", "GCC" => "Ala",
             "GCA" => "Ala", "GCG" => "Ala", "UAU" => "Tyr", "UAC" => "
                Tyr", "UAA" => "STP",
             "UAG" => "STP", "CAU" => "His", "CAC" => "His", "CAA" => "
                Gln", "CAG" => "Gln",
             "AAU" => "Asn", "AAC" => "Asn", "AAA" => "Lys", "AAG" => "
                Lys", "GAU" => "Asp",
             "GAC" => "Asp", "GAA" => "Glu", "GAG" => "Glu", "UGU" => "
                Cys", "UGC" => "Cys",
             "UGA" => "STP", "UGG" => "Trp", "CGU" => "Arg", "CGC" => "
                Arg", "CGA" => "Arg",
             "CGG" => "Arg", "AGU" => "Ser", "AGC" => "Ser", "AGA" => "
                Arg", "AGG" => "Arg",
             "GGU" => "Gly", "GGC" => "Gly", "GGA" => "Gly", "GGG" => "
                Gly" );

%revcodemap = ( );
foreach $key (keys %codemap) {
  push @{ $revcodemap{$codemap{$key}} }, $key;
}

# diagnostic printouts for arrays
# foreach $key (keys %codemap) {
  # print "$key => $codemap{$key}\n";
# }

# foreach $key (keys %revcodemap) {
  # print "$key =>";
  # foreach $val (@{ $revcodemap{$key} }) {
```

```
42      # print " $val";
    # }
44    # print "\n";
# }
46
open(IN, "<", $fname) or die $!;
48
%orgs_data = ( );
50
while(<IN>) {
52    chomp;
    if(/(.+):(.+): (\d+)/) {
54      $taxid = $1;
      $spname = $2;
56      $cdscount = $3;
      # print "----\n$taxid $spname $cdscount\n";
58      $orgs_data{$taxid}->{"spname"} = $spname;
      $orgs_data{$taxid}->{"cdscount"} = $cdscount;
60
      #print $orgs_data{$taxid}->{"spname"}."\n";
62      #print $orgs_data{$taxid}->{"cdscount"}."\n";
    }
64    else {
      # Split line into codons
66      @codon_counts = split( / );
      $total_codons = 0;
68      foreach $count (@codon_counts) {
        $total_codons += $count;
70      }
      if($#codons != $#codon_counts) {
72        print "Bad codon count - doesn't match labels at $taxid\n";
        exit;
74      }
      for($i=0;$i<=$#codons;$i++) {
76        my $cur_codon = $codons[$i];
        # $orgs_data{$taxid}->{"data"}->{$cur_codon} = $codon_counts[$i]/
            $total_codons;
78        # In this version, these are RAW counts - actual % within synonym
            group will be
        # be calculated next
80        $orgs_data{$taxid}->{"data"}->{$cur_codon} = $codon_counts[$i];
      }
82    }
}
84
```

```
# take big data array, calculate/save proportions
foreach $key (keys %orgs_data) {
  foreach $aa (keys %revcodemap) {
    $init = 1;
    foreach $codon (@{ $revcodemap{$aa} }) {
      if($init == 1) {
        $orgs_data{$key}->{"counts"}->{$aa} = 0;
        $init = 0;
      }
      $orgs_data{$key}->{"counts"}->{$aa} += $orgs_data{$key}->{"data"
         }->{$codon};
    }
  }
  # totals have been calculated, change numbers in {"data"} slide to
     proportions
  foreach $aa (keys %revcodemap) {
    foreach $codon (@{ $revcodemap{$aa} }) {
      if($orgs_data{$key}->{"counts"}->{$aa} == 0) {
        $orgs_data{$key}->{"data"}->{$codon} = 0;
      } else {
        $orgs_data{$key}->{"data"}->{$codon}
          = $orgs_data{$key}->{"data"}->{$codon}/$orgs_data{$key}->{"
             counts"}->{$aa};
      }
      # print("$key $aa $codon\n");
      # print($orgs_data{$key}->{"data"}->{$codon}."\n");
      # print($orgs_data{$key}->{"counts"}->{$aa}."\n");
    }
  }
}

# print out the data arrays - R format
$num = scalar(keys(%orgs_data));
$rownames = "c(";
$start = 1;
foreach $key (keys %orgs_data) {
  $name = $key;
  $name =~ /^(\d+)/;
        $name = $1;
  if($start != 1) {
    $rownames .= ",";
  } else {
    $start = 0;
  }
  $rownames .= "\"$name\"";
```

```
}
$rownames .= ")";

$colnames = "c(";
$start = 1;
foreach $codon (@codons) {
  if($start != 1) {
    $colnames .= ",";
  } else {
    $start = 0;
  }
  $colnames .= "\"$codon\"";
}
$colnames .= ")";

# Set up R matrix
print("gBiasData <- matrix(nrow=$num,ncol=64,dimnames=list($rownames,
   $colnames))\n");

# ...and print data.
foreach $key (keys %orgs_data) {
  $name = $key;
  $name =~ /^(\d+)/;
        $name = $1;
  foreach $codon (@codons) {
    print("gBiasData[\"$name\",\"$codon\"] <- ".$orgs_data{$key}->{'
       data'}->{$codon}."\n");
  }
}

#foreach $key (keys %orgs_data) {
  #$name = $key;
  #$name =~ s/ /_/g;
  #$name =~ s/:/_/g;
  #$name =~ s/'//g;
  #$name =~ s/,//g;
  #$name =~ s/\.//g;
  #print "BiasData.Msp\$tax".$name." <- array(c(".
#$orgs_data{$key}->{'data'}->{'AAA'}.",".$orgs_data{$key}->{'data'}->{'
   UAA'}.",".$orgs_data{$key}->{'data'}->{'GAA'}.",\n".
#$orgs_data{$key}->{'data'}->{'CAA'}.",".$orgs_data{$key}->{'data'}->{'
   AUA'}.",".$orgs_data{$key}->{'data'}->{'UUA'}.",\n".
#$orgs_data{$key}->{'data'}->{'GUA'}.",".$orgs_data{$key}->{'data'}->{'
   CUA'}.",".$orgs_data{$key}->{'data'}->{'AGA'}.",\n".
```

```
166 #$orgs_data{$key}->{'data'}->{'UGA'}.",".$orgs_data{$key}->{'data'}->{'
    GGA'}.",".$orgs_data{$key}->{'data'}->{'CGA'}.",\n".
    #$orgs_data{$key}->{'data'}->{'ACA'}.",".$orgs_data{$key}->{'data'}->{'
    UCA'}.",".$orgs_data{$key}->{'data'}->{'GCA'}.",\n".
168 #$orgs_data{$key}->{'data'}->{'CCA'}.",".$orgs_data{$key}->{'data'}->{'
    AAU'}.",".$orgs_data{$key}->{'data'}->{'UAU'}.",\n".
    #$orgs_data{$key}->{'data'}->{'GAU'}.",".$orgs_data{$key}->{'data'}->{'
    CAU'}.",".$orgs_data{$key}->{'data'}->{'AUU'}.",\n".
170 #$orgs_data{$key}->{'data'}->{'UUU'}.",".$orgs_data{$key}->{'data'}->{'
    GUU'}.",".$orgs_data{$key}->{'data'}->{'CUU'}.",\n".
    #$orgs_data{$key}->{'data'}->{'AGU'}.",".$orgs_data{$key}->{'data'}->{'
    UGU'}.",".$orgs_data{$key}->{'data'}->{'GGU'}.",\n".
172 #$orgs_data{$key}->{'data'}->{'CGU'}.",".$orgs_data{$key}->{'data'}->{'
    ACU'}.",".$orgs_data{$key}->{'data'}->{'UCU'}.",\n".
    #$orgs_data{$key}->{'data'}->{'GCU'}.",".$orgs_data{$key}->{'data'}->{'
    CCU'}.",".$orgs_data{$key}->{'data'}->{'AAG'}.",\n".
174 #$orgs_data{$key}->{'data'}->{'UAG'}.",".$orgs_data{$key}->{'data'}->{'
    GAG'}.",".$orgs_data{$key}->{'data'}->{'CAG'}.",\n".
    #$orgs_data{$key}->{'data'}->{'AUG'}.",".$orgs_data{$key}->{'data'}->{'
    UUG'}.",".$orgs_data{$key}->{'data'}->{'GUG'}.",\n".
176 #$orgs_data{$key}->{'data'}->{'CUG'}.",".$orgs_data{$key}->{'data'}->{'
    AGG'}.",".$orgs_data{$key}->{'data'}->{'UGG'}.",\n".
    #$orgs_data{$key}->{'data'}->{'GGG'}.",".$orgs_data{$key}->{'data'}->{'
    CGG'}.",".$orgs_data{$key}->{'data'}->{'ACG'}.",\n".
178 #$orgs_data{$key}->{'data'}->{'UCG'}.",".$orgs_data{$key}->{'data'}->{'
    GCG'}.",".$orgs_data{$key}->{'data'}->{'CCG'}.",\n".
    #$orgs_data{$key}->{'data'}->{'AAC'}.",".$orgs_data{$key}->{'data'}->{'
    UAC'}.",".$orgs_data{$key}->{'data'}->{'GAC'}.",\n".
180 #$orgs_data{$key}->{'data'}->{'CAC'}.",".$orgs_data{$key}->{'data'}->{'
    AUC'}.",".$orgs_data{$key}->{'data'}->{'UUC'}.",\n".
    #$orgs_data{$key}->{'data'}->{'GUC'}.",".$orgs_data{$key}->{'data'}->{'
    CUC'}.",".$orgs_data{$key}->{'data'}->{'AGC'}.",\n".
182 #$orgs_data{$key}->{'data'}->{'UGC'}.",".$orgs_data{$key}->{'data'}->{'
    GGC'}.",".$orgs_data{$key}->{'data'}->{'CGC'}.",\n".
    #$orgs_data{$key}->{'data'}->{'ACC'}.",".$orgs_data{$key}->{'data'}->{'
    UCC'}.",".$orgs_data{$key}->{'data'}->{'GCC'}.",\n".
184 #$orgs_data{$key}->{'data'}->{'CCC'}."),c(4,4,4),".
    #"dimnames=list(c(\"A\",\"U\",\"G\",\"C\"),c(\"A\",\"U\",\"G\",\"C\"),c
    (\"A\",\"U\",\"G\",\"C\")))\n";
186
  #print "\n\n";
188 #}
```

codon-funcs.r :

```
distfromeven_G <- function(meth)
{
EvenDistrMat <- matrix(nrow=1,ncol=64,dimnames=list(c("EvenDist"),c("
    CGA","CGC","CGG","CGU","AGA","AGG","CUA","CUC","CUG","CUU","UUA","
    UUG","UCA","UCC","UCG","UCU","AGC","AGU","ACA","ACC","ACG","ACU","
    CCA","CCC","CCG","CCU","GCA","GCC","GCG","GCU","GGA","GGC","GGG","
    GGU","GUA","GUC","GUG","GUU","AAA","AAG","AAC","AAU","CAA","CAG","
    CAC","CAU","GAA","GAG","GAC","GAU","UAC","UAU","UGC","UGU","UUC","
    UUU","AUA","AUC","AUU","AUG","UGG","UAA","UAG","UGA")))
EvenDistrMat["EvenDist","CGA"] <- 0.166666666666667
EvenDistrMat["EvenDist","CGC"] <- 0.166666666666667
EvenDistrMat["EvenDist","CGG"] <- 0.166666666666667
EvenDistrMat["EvenDist","CGU"] <- 0.166666666666667
EvenDistrMat["EvenDist","AGA"] <- 0.166666666666667
EvenDistrMat["EvenDist","AGG"] <- 0.166666666666667
EvenDistrMat["EvenDist","CUA"] <- 0.166666666666667
EvenDistrMat["EvenDist","CUC"] <- 0.166666666666667
EvenDistrMat["EvenDist","CUG"] <- 0.166666666666667
EvenDistrMat["EvenDist","CUU"] <- 0.166666666666667
EvenDistrMat["EvenDist","UUA"] <- 0.166666666666667
EvenDistrMat["EvenDist","UUG"] <- 0.166666666666667
EvenDistrMat["EvenDist","UCA"] <- 0.166666666666667
EvenDistrMat["EvenDist","UCC"] <- 0.166666666666667
EvenDistrMat["EvenDist","UCG"] <- 0.166666666666667
EvenDistrMat["EvenDist","UCU"] <- 0.166666666666667
EvenDistrMat["EvenDist","AGC"] <- 0.166666666666667
EvenDistrMat["EvenDist","AGU"] <- 0.166666666666667
EvenDistrMat["EvenDist","ACA"] <- 0.25
EvenDistrMat["EvenDist","ACC"] <- 0.25
EvenDistrMat["EvenDist","ACG"] <- 0.25
EvenDistrMat["EvenDist","ACU"] <- 0.25
EvenDistrMat["EvenDist","CCA"] <- 0.25
EvenDistrMat["EvenDist","CCC"] <- 0.25
EvenDistrMat["EvenDist","CCG"] <- 0.25
EvenDistrMat["EvenDist","CCU"] <- 0.25
EvenDistrMat["EvenDist","GCA"] <- 0.25
EvenDistrMat["EvenDist","GCC"] <- 0.25
EvenDistrMat["EvenDist","GCG"] <- 0.25
EvenDistrMat["EvenDist","GCU"] <- 0.25
EvenDistrMat["EvenDist","GGA"] <- 0.25
EvenDistrMat["EvenDist","GGC"] <- 0.25
EvenDistrMat["EvenDist","GGG"] <- 0.25
EvenDistrMat["EvenDist","GGU"] <- 0.25
EvenDistrMat["EvenDist","GUA"] <- 0.25
```

```
EvenDistrMat["EvenDist","GUC"] <- 0.25
EvenDistrMat["EvenDist","GUG"] <- 0.25
EvenDistrMat["EvenDist","GUU"] <- 0.25
EvenDistrMat["EvenDist","AAA"] <- 0.5
EvenDistrMat["EvenDist","AAG"] <- 0.5
EvenDistrMat["EvenDist","AAC"] <- 0.5
EvenDistrMat["EvenDist","AAU"] <- 0.5
EvenDistrMat["EvenDist","CAA"] <- 0.5
EvenDistrMat["EvenDist","CAG"] <- 0.5
EvenDistrMat["EvenDist","CAC"] <- 0.5
EvenDistrMat["EvenDist","CAU"] <- 0.5
EvenDistrMat["EvenDist","GAA"] <- 0.5
EvenDistrMat["EvenDist","GAG"] <- 0.5
EvenDistrMat["EvenDist","GAC"] <- 0.5
EvenDistrMat["EvenDist","GAU"] <- 0.5
EvenDistrMat["EvenDist","UAC"] <- 0.5
EvenDistrMat["EvenDist","UAU"] <- 0.5
EvenDistrMat["EvenDist","UGC"] <- 0.5
EvenDistrMat["EvenDist","UGU"] <- 0.5
EvenDistrMat["EvenDist","UUC"] <- 0.5
EvenDistrMat["EvenDist","UUU"] <- 0.5
EvenDistrMat["EvenDist","AUA"] <- 0.5
EvenDistrMat["EvenDist","AUC"] <- 1
EvenDistrMat["EvenDist","AUU"] <- 0.5
EvenDistrMat["EvenDist","AUG"] <- 1
EvenDistrMat["EvenDist","UGG"] <- 1
EvenDistrMat["EvenDist","UAA"] <- 0.333333333333333
EvenDistrMat["EvenDist","UAG"] <- 0.333333333333333
EvenDistrMat["EvenDist","UGA"] <- 0.333333333333333

  mats <- row.names(gBiasData)
  len <- length(mats)

  if(len < 2)
    stop("Not enough data")

  d <- matrix(nrow=len,ncol=1)

  for(x in c(1:len)) {
    tmp <- matrix(nrow=2,ncol=64)
    tmp[1,] <- EvenDistrMat[1,]
    tmp[2,] <- gBiasData[x,]
    d[x] <- dist(tmp,method=meth)
  }
```

```
84    row.names(d) <- mats

86    return(d)
}
88
# Turn a list of 64-col vectors into first k singular vectors
90 k_svd <- function(x,k)
{
92    len <- nrow(x)
      s <- matrix(nrow=len,ncol=k)
94    row.names(s) <- row.names(x)

96    for(n in c(1:len)) {
        t.svd <- svd(x[n,])
98      s[n,] <- t.svd$v[,1] %*% t.svd$d[1] %*% t(t.svd$u)[,1:k]
      }
100
      return(s)
102 }

104 # Cosine similarity between vectors X and Y
    # http://en.wikipedia.org/wiki/Cosine_similarity
106 cos_sim <- function(x,y)
    {
108         sim <- (x %*% y) / ( (sqrt(sum(x^2)))*(sqrt(sum(y^2))) )
            return(sim)
110 }

112 # average variance of the rows in data set
    a_var <- function(x)
114 {
            len <- nrow(x)
116         s <- matrix(nrow=len,ncol=1)
            for(n in c(1:len)) {
118                 s[n,] <- var(x[n,])
            }
120         return(mean(s))
    }
```

compare_trees.pl :

```
1 #!/usr/bin/perl
  #
3 # Take two trees & annotate tree 1 re: tree 2
```

```
#

use Bio::Phylo::IO qw(parse unparse);

$tree_1 = $ARGV[0];
if(! -f $tree_1) {
  die("Please provide an input file for tree 1.\n");
}

$tree_2 = $ARGV[1];
if(! -f $tree_2) {
  die("Please provide an input file for tree 2.\n");
}

# load trees from file
my $tree_first = parse(
  '-file' => $tree_1,
  '-format' => 'newick',
)->first;

my $tree_second = parse(
  '-file' => $tree_2,
  '-format' => 'newick',
)->first;



#$tree_first->visit_depth_first(
  #'-in' => sub { my $name = shift->get_name(); if($name != "") { print
      "in: ",$name,"\n" } },
#);

# walk the first tree, and check for similar adjacency in the second
    tree.
@first_terminals = @{ $tree_first->get_terminals() };
#@first_terminals = fix_node_names(@first_terminals);
fix_node_names(@first_terminals);
@second_terminals = @{ $tree_second->get_terminals() };
#@second_terminals = fix_node_names(@second_terminals);
fix_node_names(@second_terminals);

%common = ();
foreach $first_node (@first_terminals) {
  # get a node collection
  @first_set = @{ $first_node->get_sisters() };
```

```
47    @first_set = fix_node_collection("nt",@first_set);
      @first_set = filter_dist($first_node,@first_set);
49    #@first_set = fix_node_names(@first_set);
      $first_name = $first_node->get_name();
51
      # find node in the second tree by the first name
53    $found = 0;
      foreach $second_node (@second_terminals) {
55      if($first_name == $second_node->get_name()) {
          $second_target = $second_node;
57        $found = 1;
          last;
59      }
      }
61    if($found == 0) {
        print STDERR "No match found for outer node $first_name\n";
63      next;
      }
65
      # compare similar node collection
67    @second_set = @{ $second_target->get_sisters() };
      @second_set = fix_node_collection("nt",@second_set);
69    @second_set = filter_dist($second_target,@second_set);
      #@second_set = fix_node_names(@second_set);
71
      $first_size = scalar(@first_set);
73    $second_size = scalar(@second_set);
      print STDERR "fs: $first_size ss: $second_size\n";
75    $common{$first_name} = 0;
      foreach $first_set_node (@first_set) {
77      $fsn_name = $first_set_node->get_name();
        #if($fsn_name ne $first_name) {
79        print STDERR "$fsn_name $first_name\n";
          foreach $second_set_node (@second_set) {
81          $ssn_name = $second_set_node->get_name;
            #print STDERR "\t$ssn_name\n";
83          if($ssn_name eq $fsn_name) {
              $common{$first_name}++;
85          }
          }
87      #}
      }
89  }

91  foreach $name (keys %common) {
```

```
    print "node␣$name␣:␣",$common{$name},"\n";
}

sub fix_node_collection {
    # first argument tells us if we're stripping non-terminal
    # nodes ("nt"), or finding and returning the full set of
    # terminals ("t")
    my $term = shift;
    my @collection = @_;
    for($i = 0; $i <= $#collection; $i++) {
        if($collection[$i]->is_internal()) {
            if($term eq "t") {
                @term_set = @{ $collection[$i]->get_terminals() };
                splice(@collection,$i,1,@term_set);
            } elsif($term eq "nt") {
                splice(@collection,$i,1);
            }
        }
    }
    return @collection;
}

sub filter_dist {
    # Returns an array of 3 closest nodes from a list
    # given a source and a list of comparison targets
    my $node = shift;
    my @set = @_;
    my $dist1 = 999999;
    my $dist2 = 999999;
    my $dist3 = 999999;

    foreach $tgt (@set) {
        print STDERR "TGT:".$tgt->get_name()."\n";
        $tmp_d = $node->calc_patristic_distance($tgt);
        if($tmp_d < $dist1 && $tmp_d < $dist2) {
            $dist2 = $dist1;
            $node2 = $node1;
            $node1 = $tgt;
            $dist1 = $tmp_d;
        }
        if($tmp_d > $dist1 && $tmp_d < $dist2) {
            $dist3 = $dist2;
            $node3 = $node2;
            $node2 = $tgt;
            $dist2 = $tmp_d;
```

```
137     }
        if($tmp_d > $dist1 && $tmp_d > $dist2 && $tmp_d < $dist3) {
139       $node3 = $tgt;
          $dist3 = $tmp_d;
141     }
        #print "D1:$dist1\n";
143     #print "D2:$dist2\n";
    }
145 @ret = ( );
    if($dist1 != 999999) { push @ret,$node1; }
147 if($dist2 != 999999) { push @ret,$node2; }
    if($dist3 != 999999) { push @ret,$node3; }
149 return @ret;
}
151
sub fix_node_names {
153 #my @ret = ( );
    #my @collection = @_;
155 foreach $node (@_) {
      $name = $node->get_name();
157   $name =~ s/'//g;
␣␣␣␣$name␣=~␣s/[_]+/_/g;
159 ␣␣␣␣$name␣=~␣s/^[_]//g;
␣␣␣␣@nameparts␣=␣split(/_/,$name);
161 ␣␣␣␣$name␣=␣$nameparts[0]."_".$nameparts[1];
␣␣␣␣print␣"$name\n";
163 ␣␣␣␣$node->set_name($name);
␣␣␣␣#push␣@ret,@node;
165 ␣␣}
␣␣#return␣@ret;
167 }
```